

Unidade IV:

Tipos Abstratos de Dados Lineares - Fila



PUC Minas

Instituto de Ciências Exatas e Informática
Departamento de Ciência da Computação

Agenda

- Conceitos Básicos
- Implementação Circular em Java
- Implementação Circular em C

Agenda

- **Conceitos Básicos** 
- Implementação Circular em Java
- Implementação Circular em C

Introdução

- As filas são um Tipo Abstrato de Dados (TAD) no qual o **primeiro elemento que entra é o primeiro a sair**
- ***First In, First Out (FIFO)***
- Tem basicamente os métodos de inserir (*enfileirar, enqueue*) e remover (*desenfileirar, dequeue*)

Exemplos



Exercício

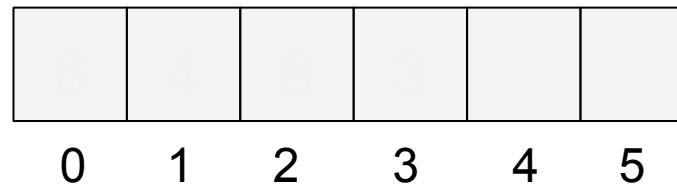
- Dado o código da lista (métodos II, IF, I, RI, RF e R), como podemos alterá-lo para criarmos uma fila? Apresente as duas soluções possíveis e mostre a desvantagem de cada uma

Exercício

- Primeira solução IF e RI (**remoção não é eficiente**)
- Segunda solução II e RF (**inserção não é eficiente**)

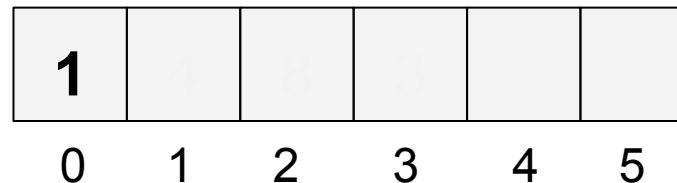
Exercício

- Primeira solução IF e RI (remoção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (inserção não é eficiente)



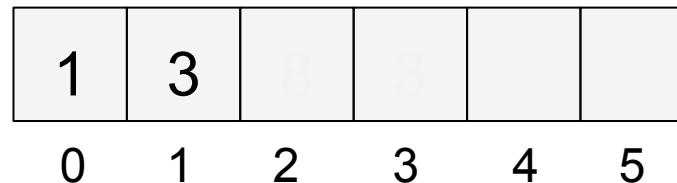
Exercício

- Primeira solução IF e RI (remoção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (inserção não é eficiente)



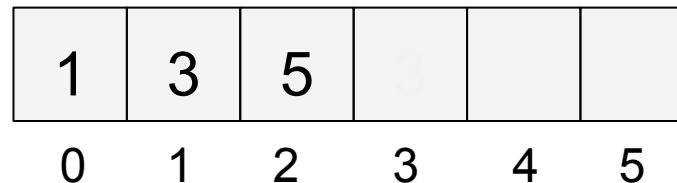
Exercício

- Primeira solução IF e RI (remoção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (inserção não é eficiente)



Exercício

- Primeira solução IF e RI (remoção não é eficiente)
 - Por exemplo, inserindo o 1, 3, **5** e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (inserção não é eficiente)



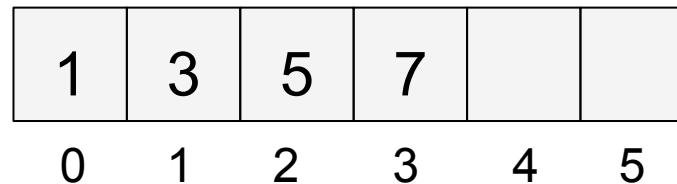
Exercício

- Primeira solução IF e RI (remoção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (inserção não é eficiente)

1	3	5	7		
0	1	2	3	4	5

Exercício

- Primeira solução IF e RI (**remoção não é eficiente**)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução

Primeira remoção: Retorna o 1 e move todos os demais

Exercício

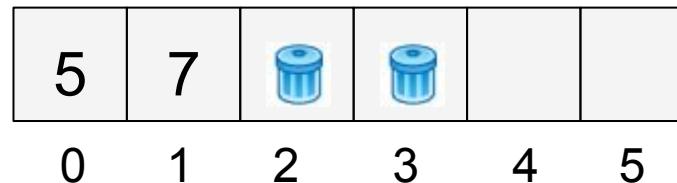
- Primeira solução IF e RI (**remoção não é eficiente**)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (inserção não é eficiente)



Primeira remoção: Retorna o 1 e move todos os demais

Exercício

- Primeira solução IF e RI (**remoção não é eficiente**)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (inserção não é eficiente)

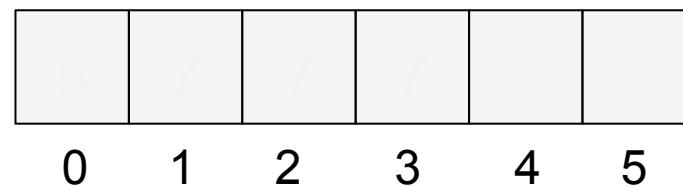


Primeira remoção: Retorna o 1 e move todos os demais

Segunda remoção: Retorna o 3 e move todos os demais

Exercício

- Primeira solução IF e RI (remoção não é eficiente)
- Segunda solução II e RF (inserção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:

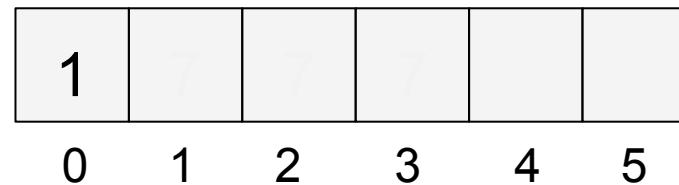


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Cada inserção: Move todos os elementos já cadastrados

- Segunda solução II e RF (inserção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:

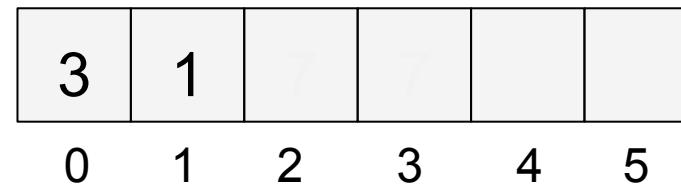


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

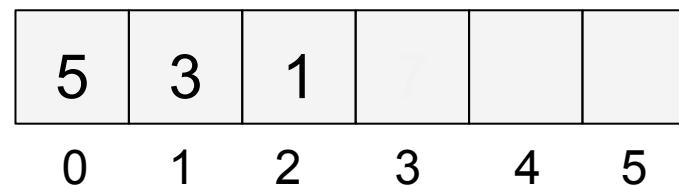
Cada inserção: Move todos os elementos já cadastrados

- Segunda solução II e RF (**inserção não é eficiente**)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



Exercício

- Primeira solução IF e RI (remoção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:
- Segunda solução II e RF (**inserção não é eficiente**)
 - Por exemplo, inserindo o 1, 3, **5** e 7 e efetuando duas remoções teremos:

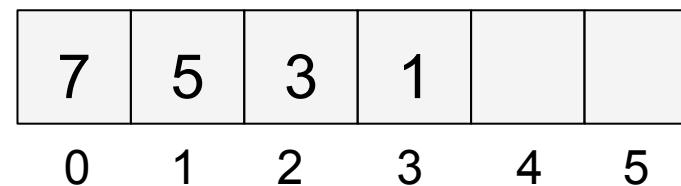


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Cada inserção: Move todos os elementos já cadastrados

- Segunda solução II e RF (**inserção não é eficiente**)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Na primeira remoção,
retiramos o número 1

- Segunda solução II e RF (inserção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:

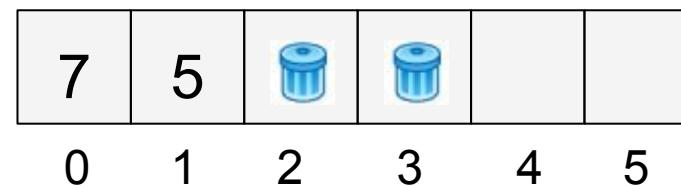


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Na segunda remoção,
retiramos o número 3

- Segunda solução II e RF (inserção não é eficiente)
 - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?

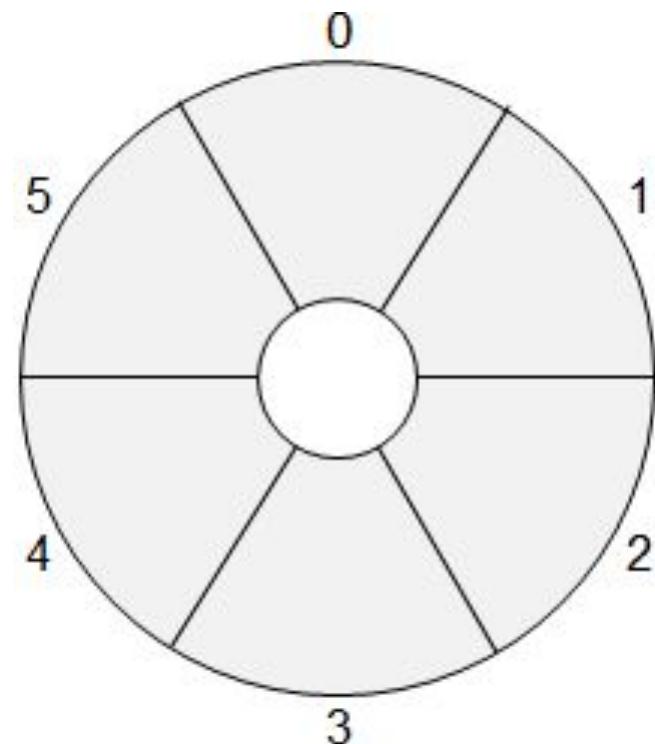
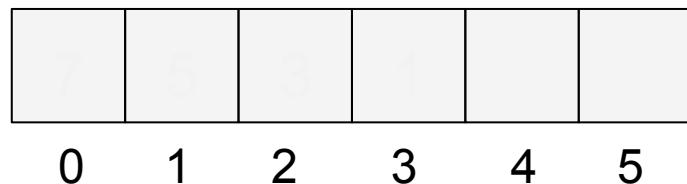
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?

E agora José?

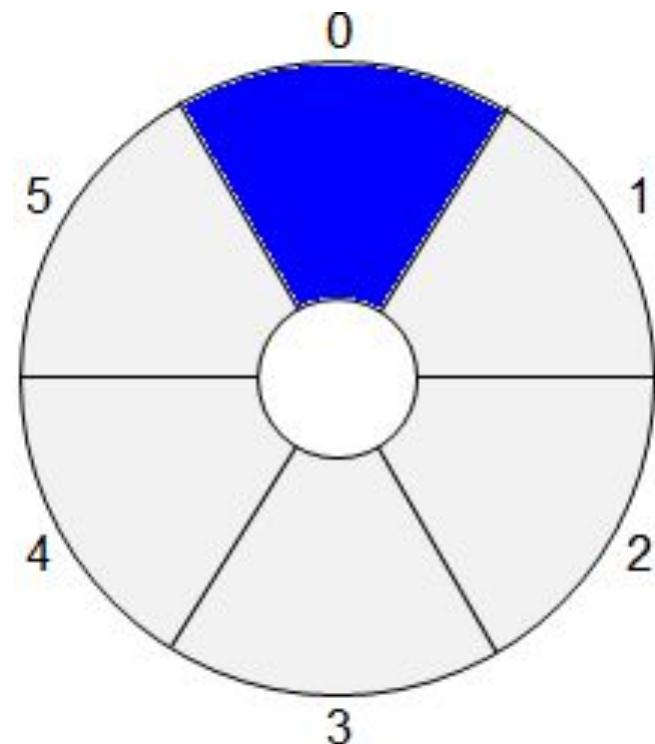
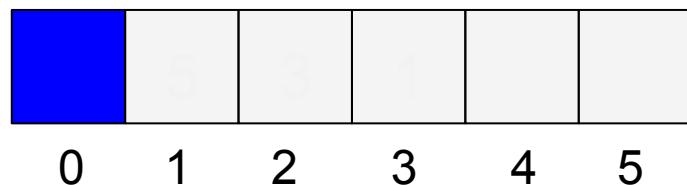
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



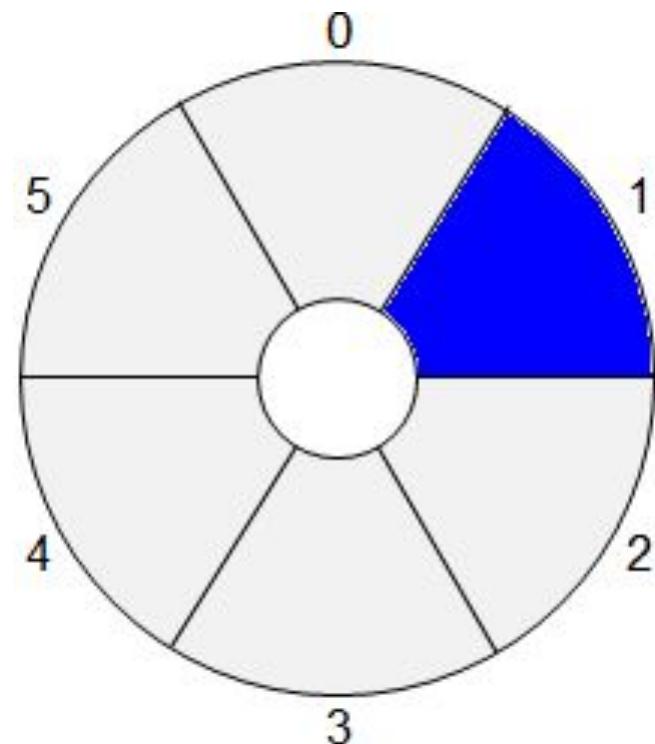
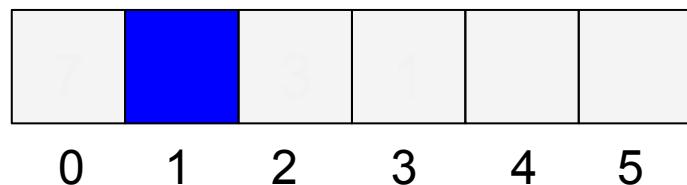
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



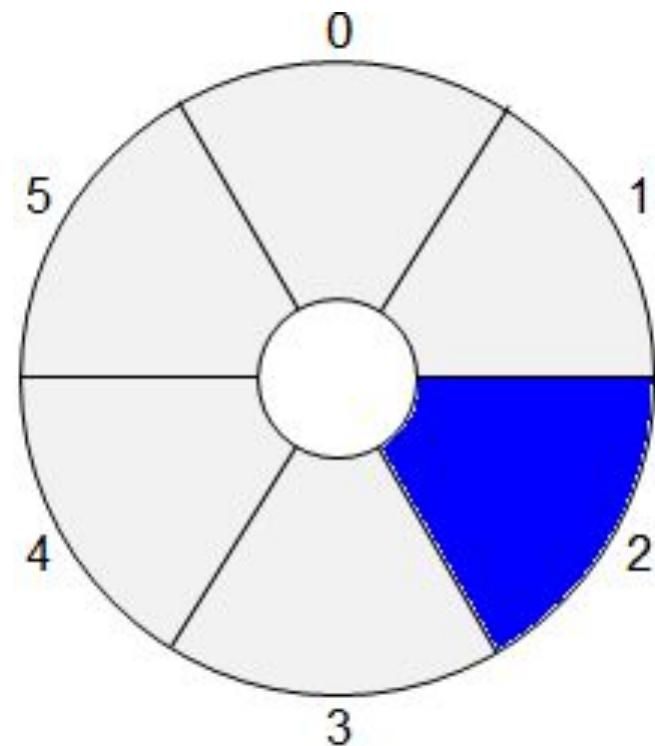
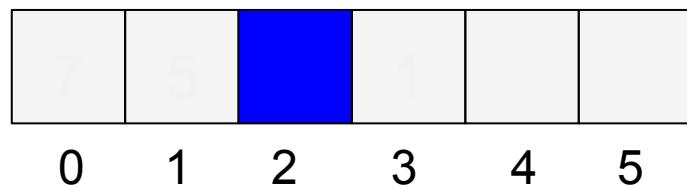
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



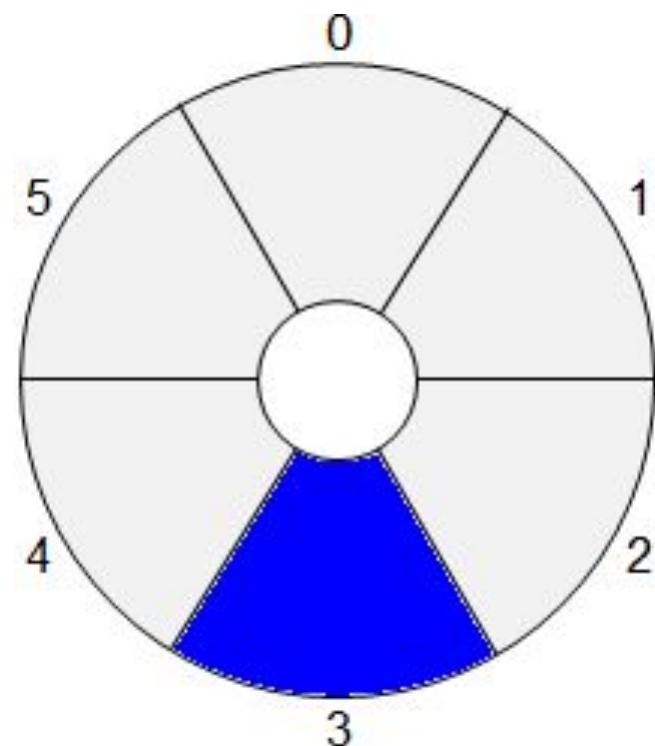
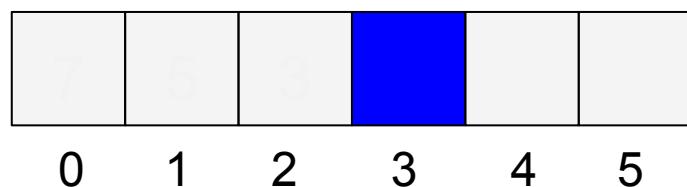
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



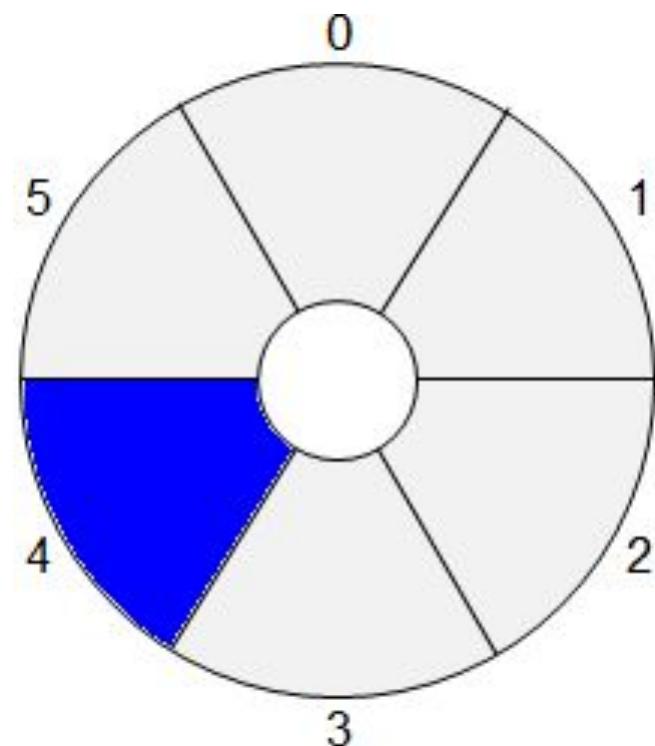
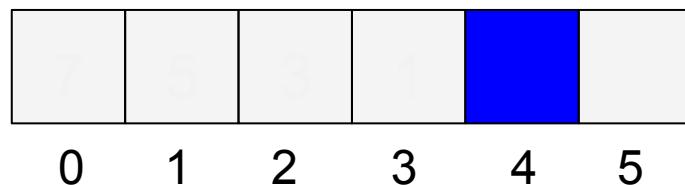
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



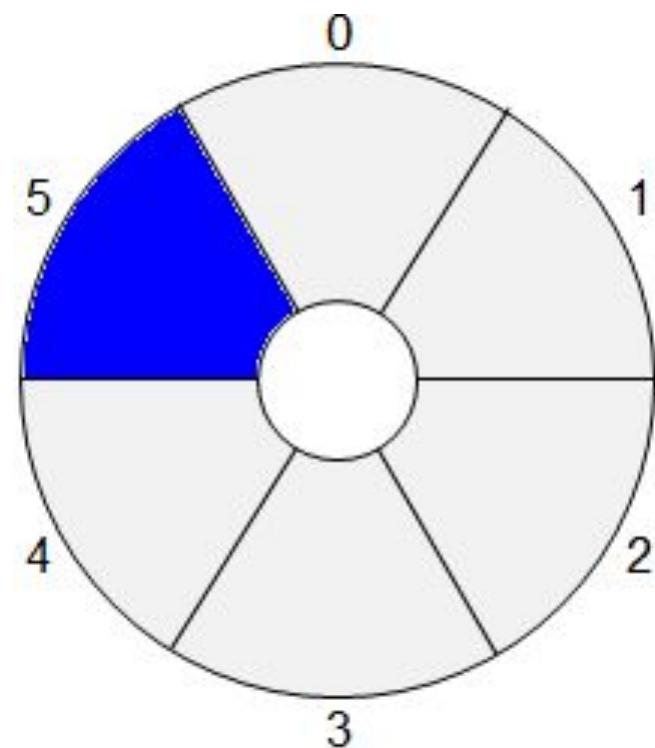
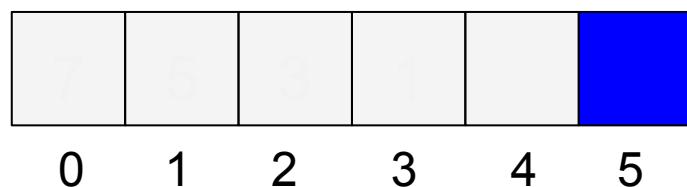
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



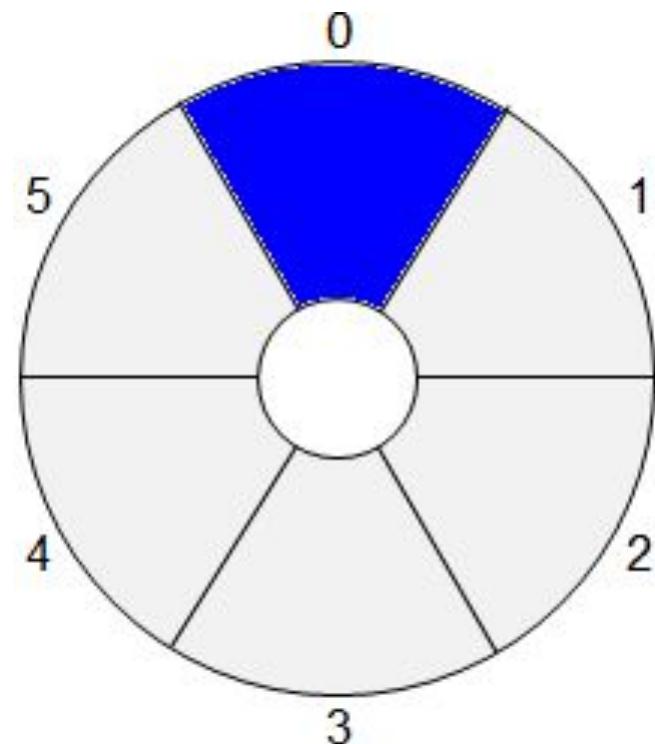
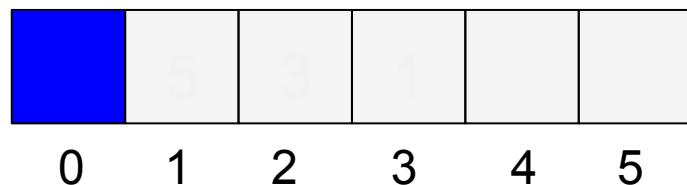
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



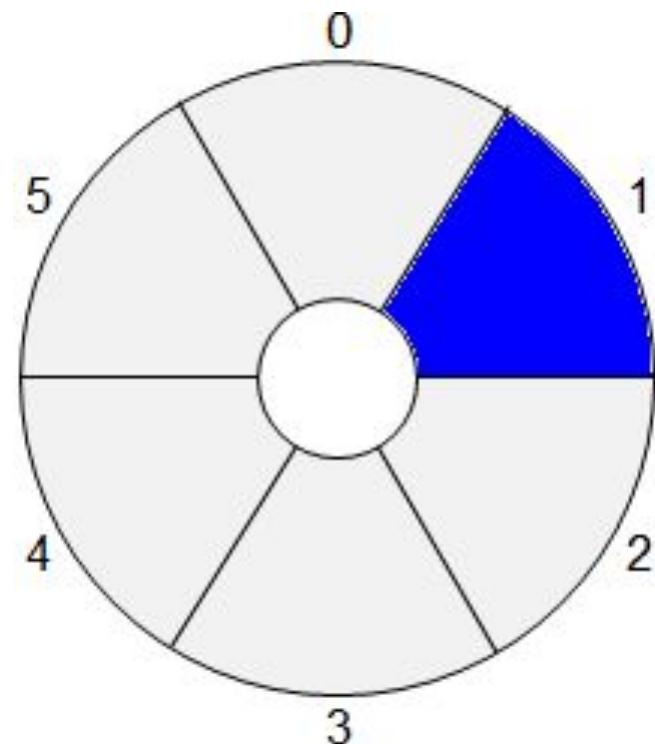
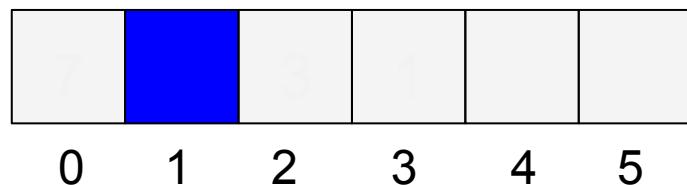
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



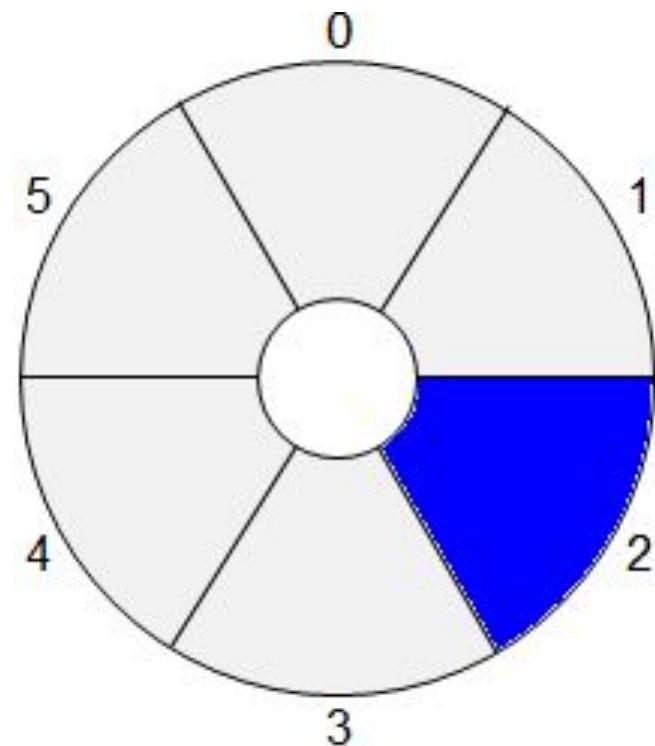
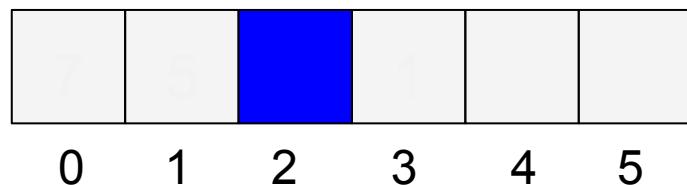
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



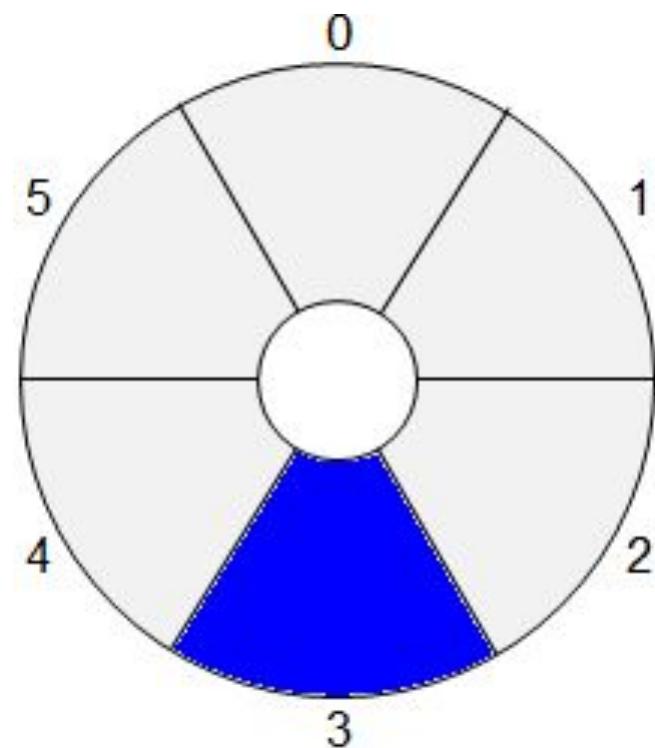
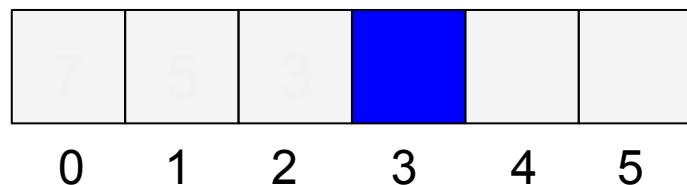
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



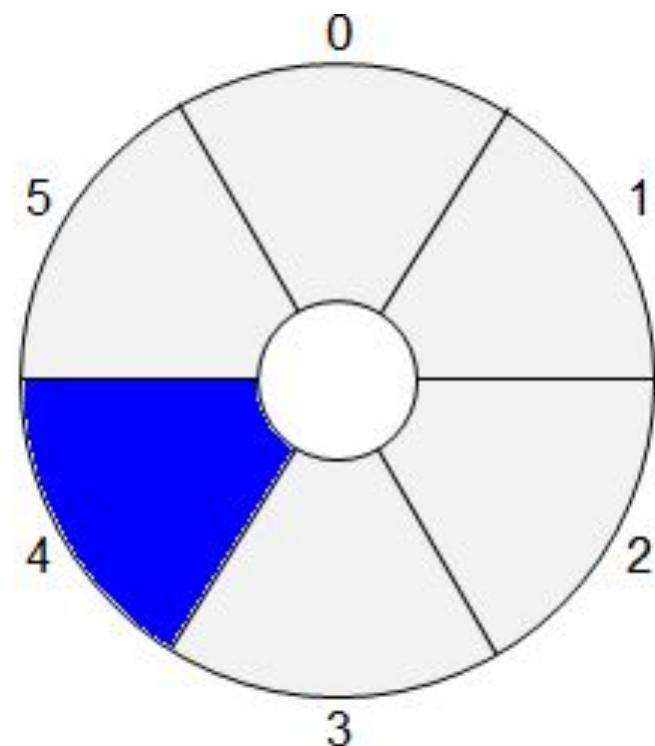
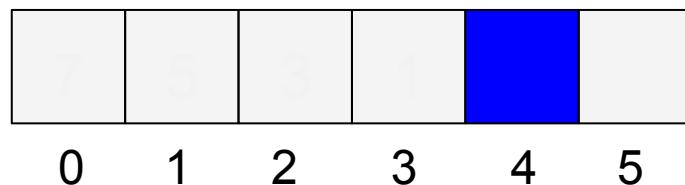
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



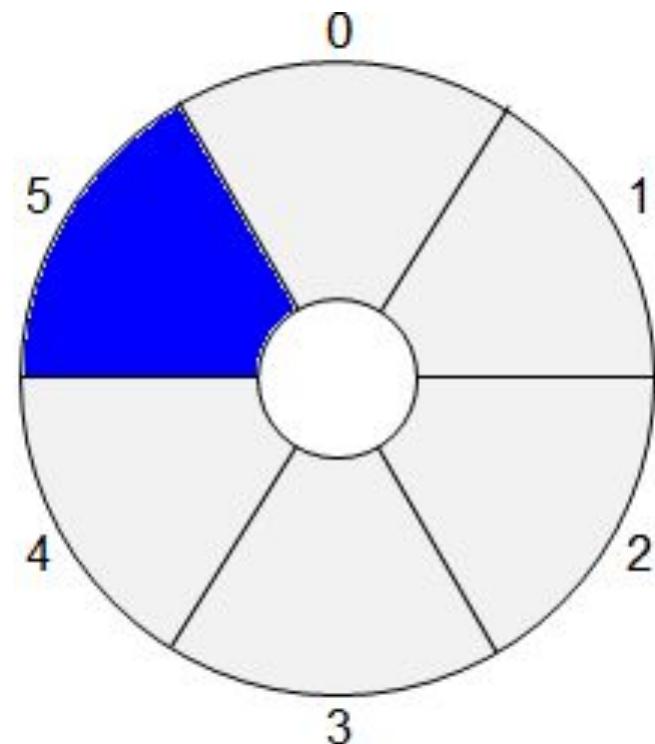
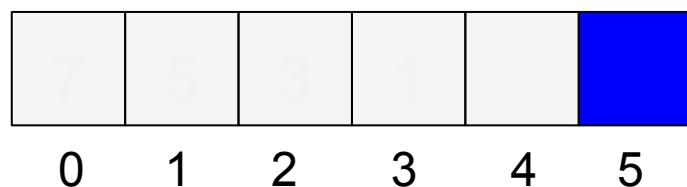
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



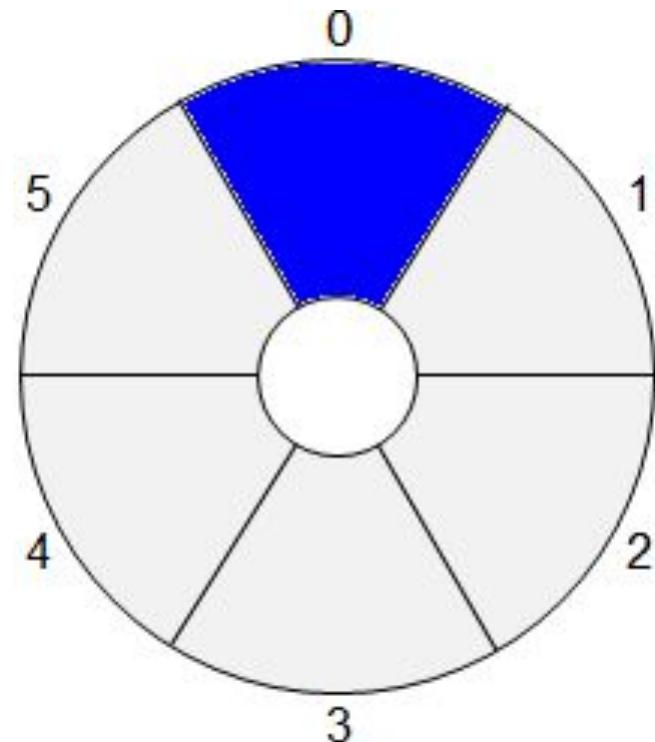
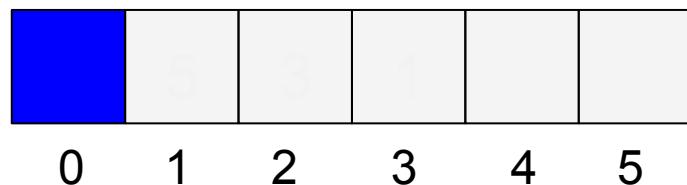
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
 - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira



Exercício

- 0 % 5 =
- 1 % 5 =
- 2 % 5 =
- 3 % 5 =
- 4 % 5 =

Exercício

- $0 \% 5 = 0$
- $1 \% 5 = 1$
- $2 \% 5 = 2$
- $3 \% 5 = 3$
- $4 \% 5 = 4$

Exercício

- Faça o quadro de memória do programa abaixo

```
n = 0;  
n = (n + 1) % 5;  
n = (n + 1) % 5;
```

Agenda

- Conceitos Básicos
- **Implementação Circular em Java** ←
- Implementação Circular em C

Algoritmo em Java

```
class Fila {  
    int[] array;  
    int primeiro, ultimo;  
  
    Fila () {  
        this(5);  
    }  
    Fila (int tamanho) {  
        array = new int[tamanho+1];  
        primeiro = ultimo = 0;  
    }  
    void inserir(int x) { ... }  
    int remover() { ... }  
    void mostrar() { ... }  
}
```

Algoritmo em Java

```
class Fila {  
    int[] array;  
    int primeiro, ultimo;  
  
    Fila () {  
        this(5);  
    }  
    Fila (int tamanho) {  
        array = new int[tamanho+1];  
        primeiro = ultimo = 0;  
    }  
    void inserir(int x) { ... }  
    int remover() { ... }  
    void mostrar() { ... }  
}
```

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

```
class Fila {  
    int[] array;  
    int primeiro, ultimo;  
  
    Fila () {  
        this(5);  
    }  
    Fila (int tamanho) {  
        array = new int[tamanho+1];  
        primeiro = ultimo = 0;  
    }  
    void inserir(int x) { ... }  
    int remover() { ... }  
    void mostrar() { ... }  
}
```

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

```
class Fila {  
    int[] array;  
    int primeiro, ultimo;  
  
    Fila () {  
        this(5);  
    }  
    Fila (int tamanho) {  
        array = new int[tamanho+1];  
        primeiro = ultimo = 0;  
    }  
    void inserir(int x) { ... }  
    int remover() { ... }  
    void mostrar() { ... }  
}
```

Vamos reservar uma unidade a mais, contudo, nossa fila caberá somente a quantidade solicitada

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

```

class Fila {
    int[] array;
    int primeiro, ultimo;

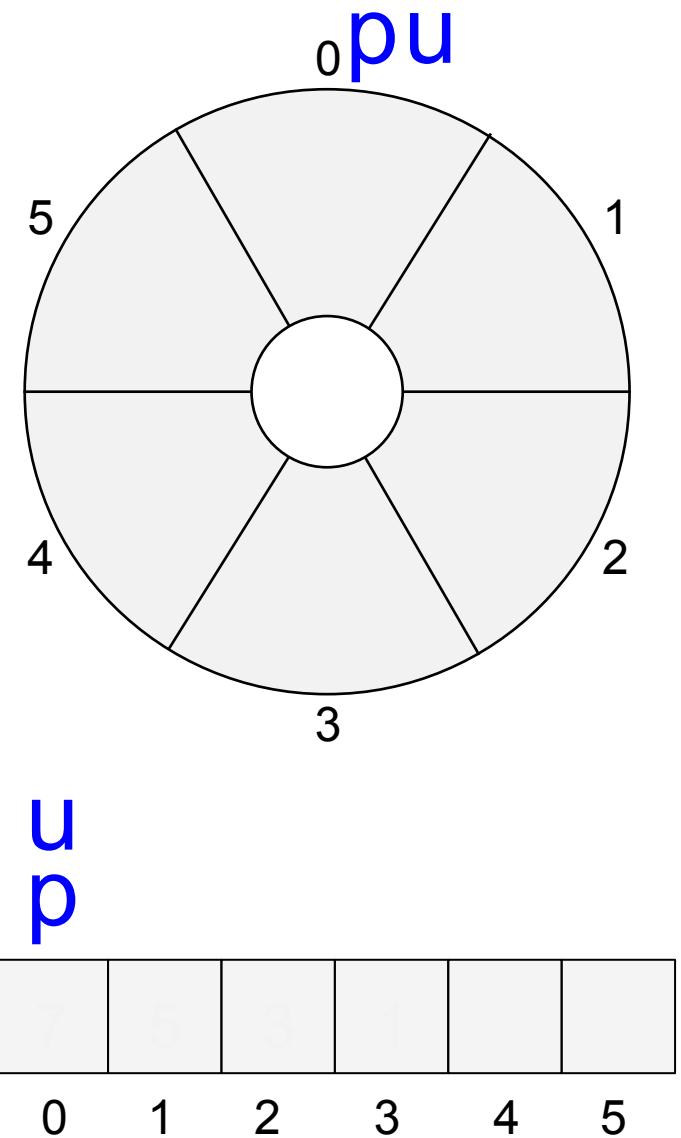
    Fila () {
        this(5);
    }

    Fila (int tamanho) {
        array = new int[tamanho+1];
        primeiro = ultimo = 0;
    }

    void inserir(int x) { ... }
    int remover() { ... }
    void mostrar() { ... }
}

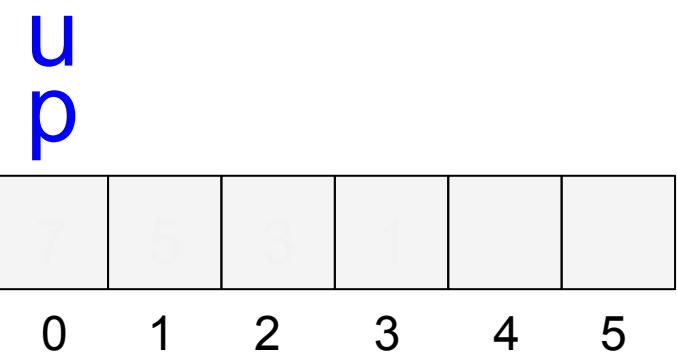
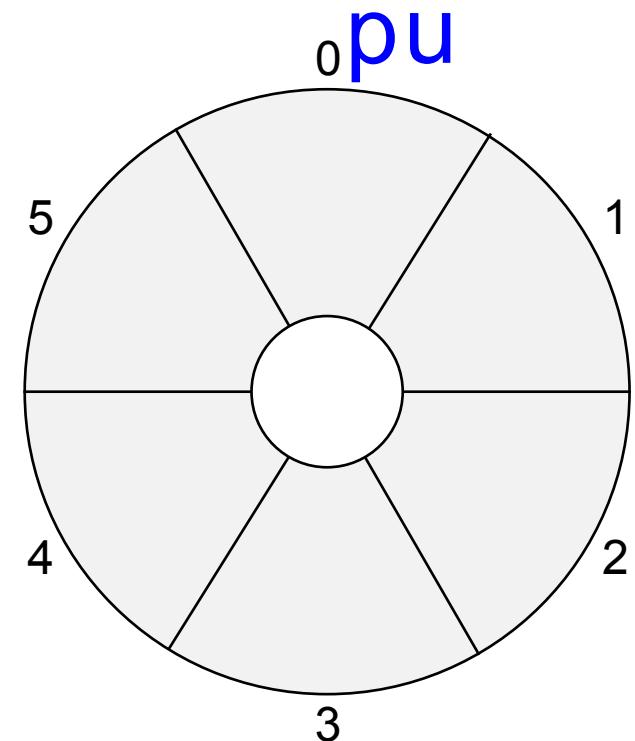
```

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```
//Inserir(1)  
  
void inserir(int x) throws Exception {  
  
    if (((ultimo + 1) % array.length) == primeiro)  
        throw new Exception("Erro!");  
  
    array[ultimo] = x;  
    ultimo = (ultimo + 1) % array.length;  
}
```



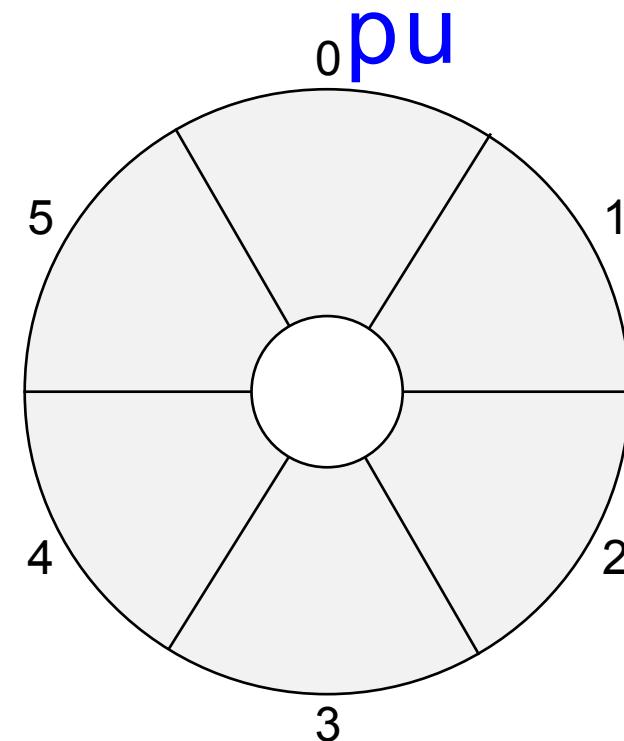
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

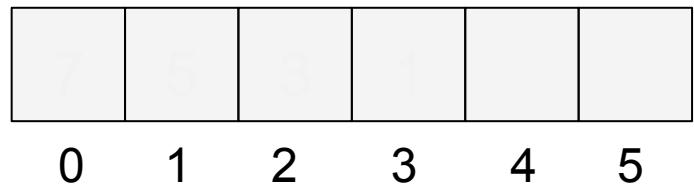
```
//Inserir(1)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");
    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}

false: 0 + 1 % 6 == 0
```



up



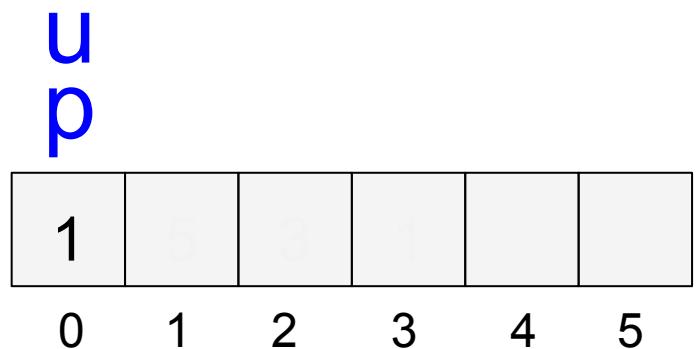
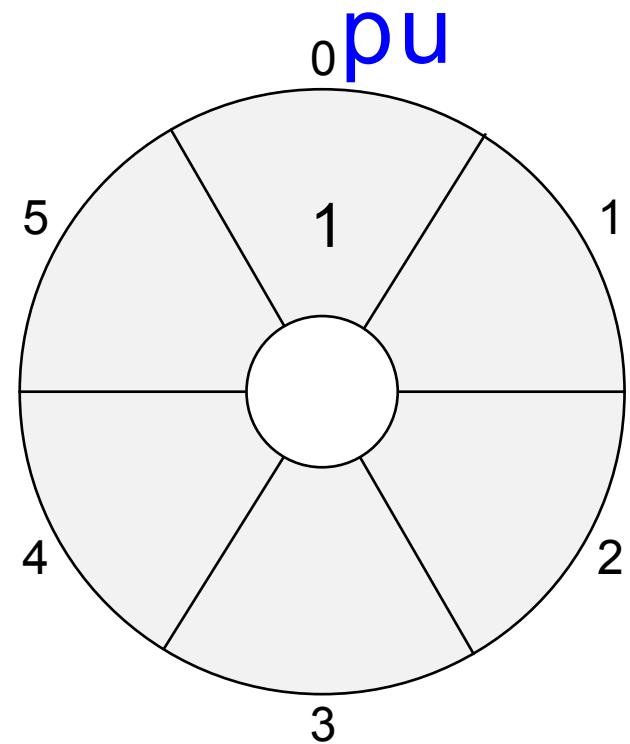
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

```
//Inserir(1)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



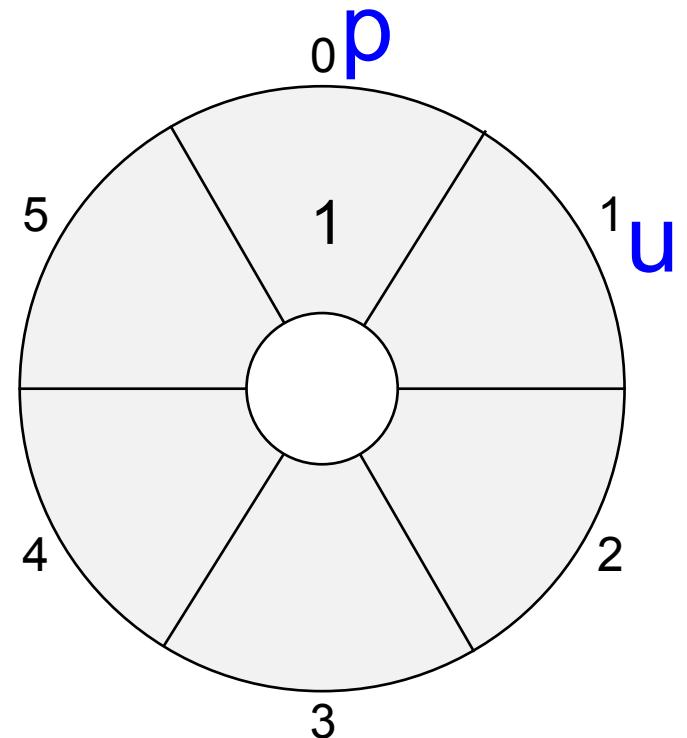
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

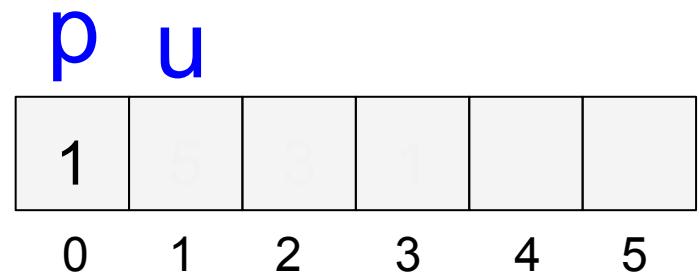
```
//Inserir(1)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

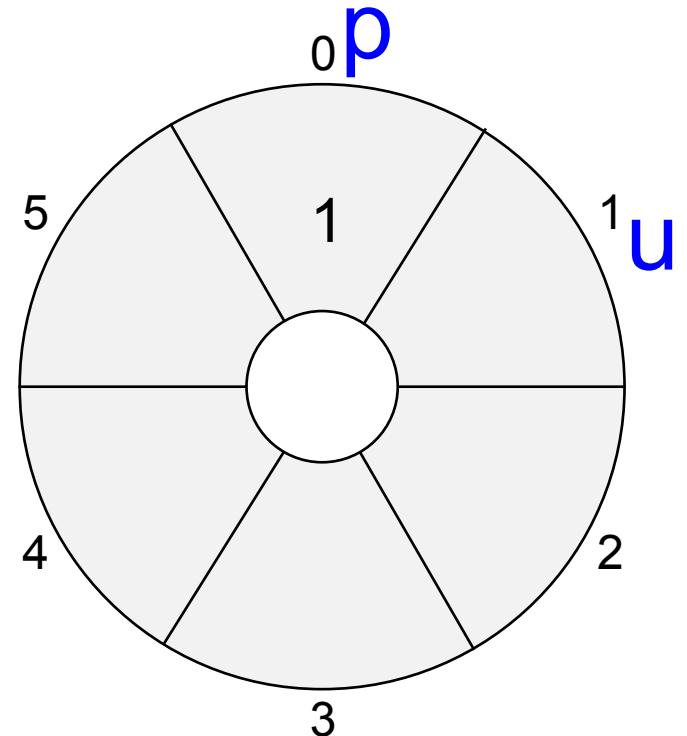


Algoritmo em Java

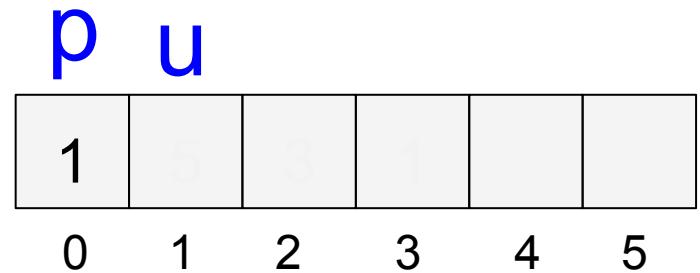
```
//Inserir(1)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



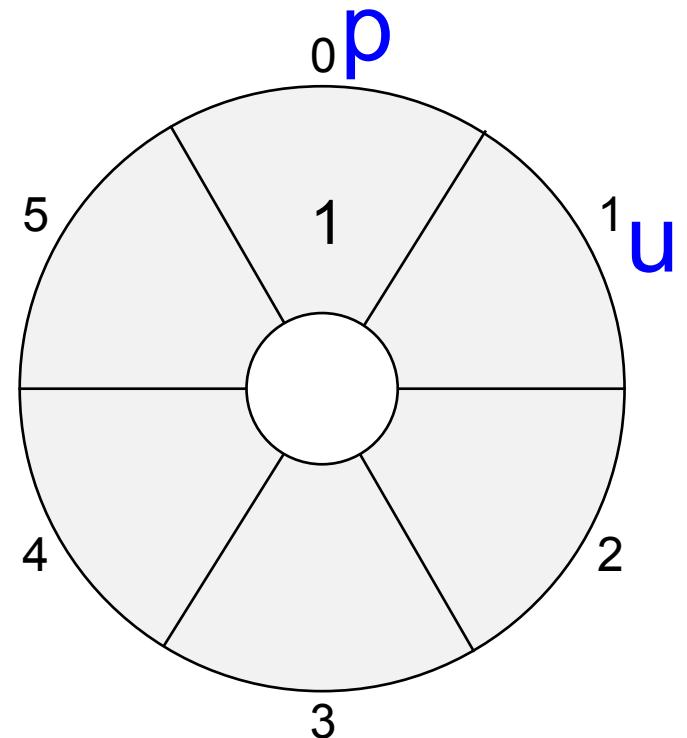
Algoritmo em Java

```
//Inserir(3)

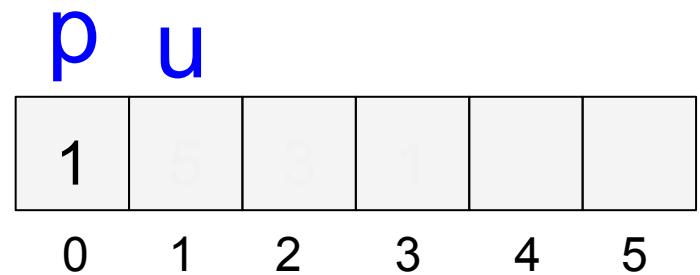
void inserir(int x) throws Exception {

    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



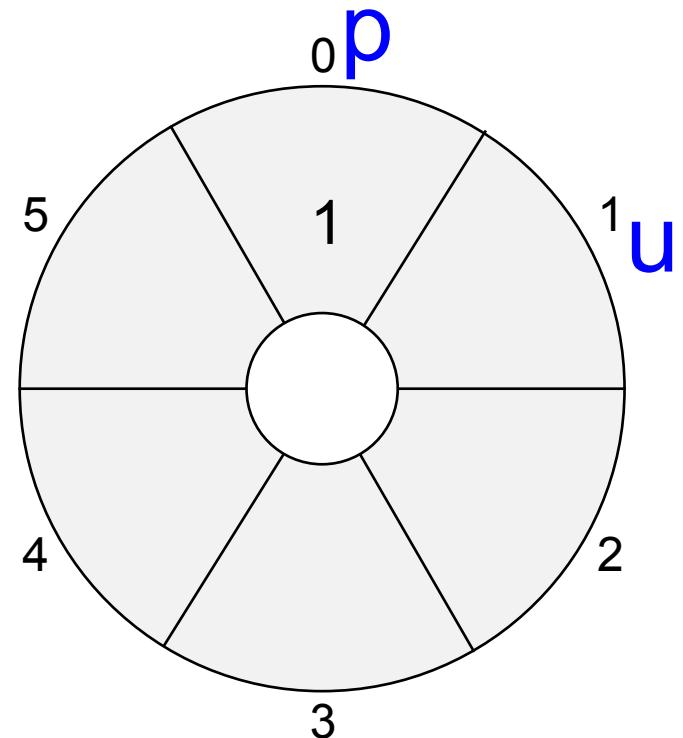
Algoritmo em Java

```
//Inserir(3)

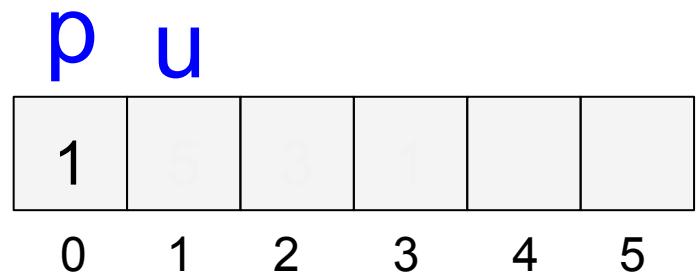
void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}

false: 1 + 1 % 6 == 0
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

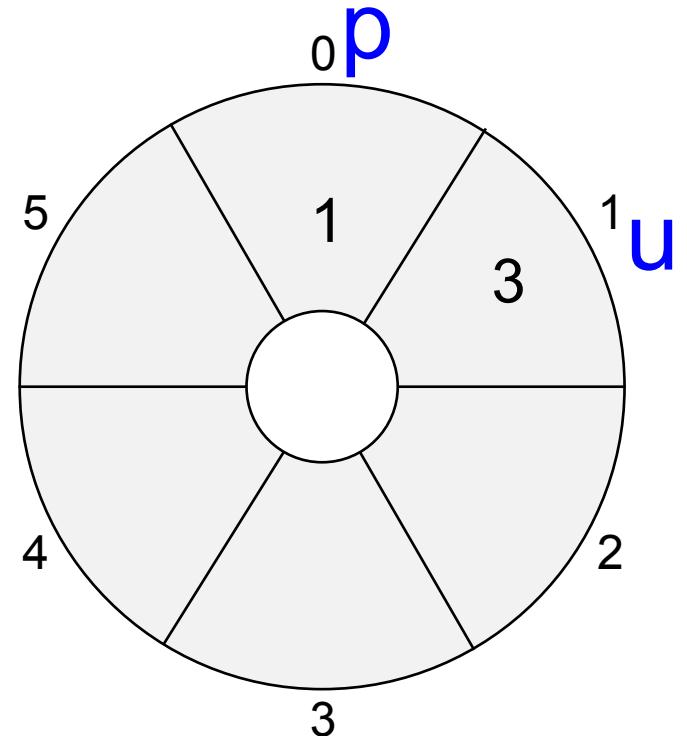


Algoritmo em Java

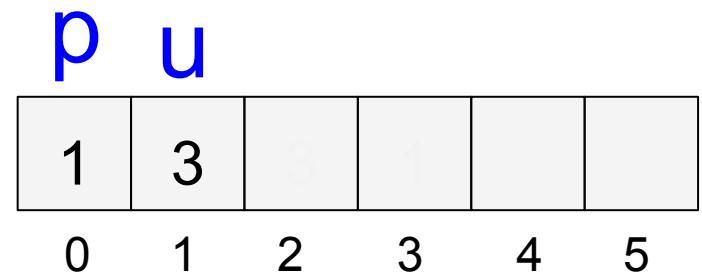
```
//Inserir(3)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

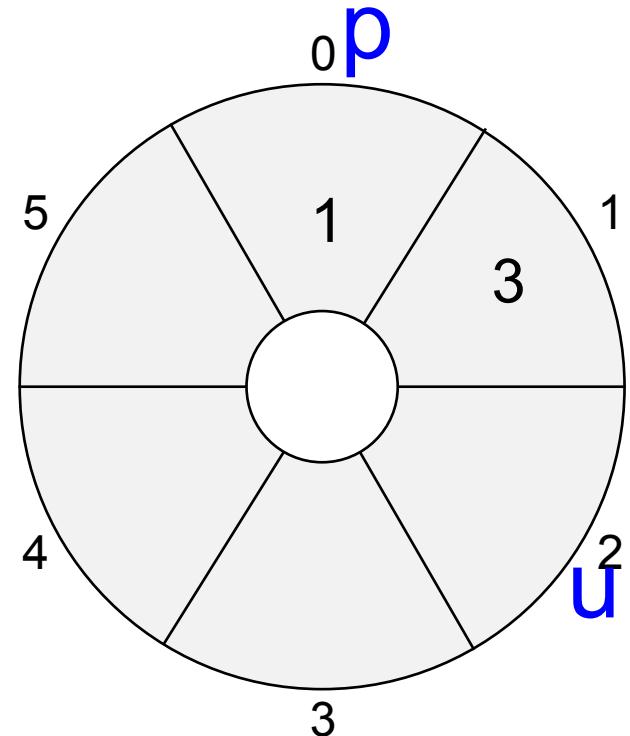


Algoritmo em Java

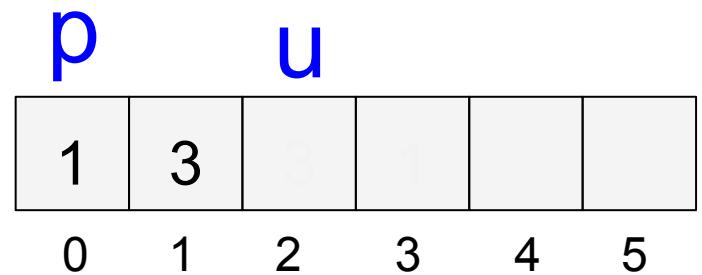
```
//Inserir(3)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

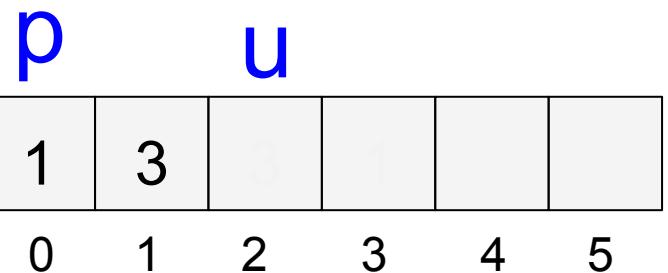
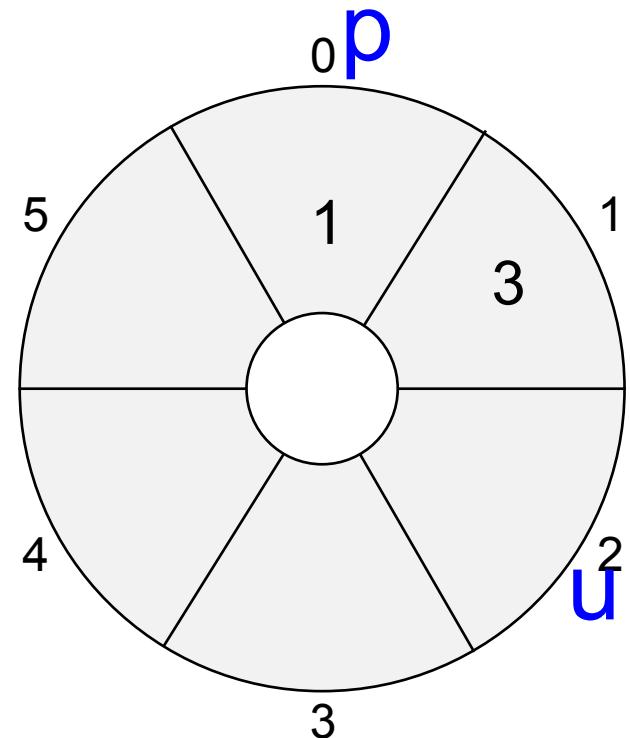


Algoritmo em Java

```
//Inserir(3)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

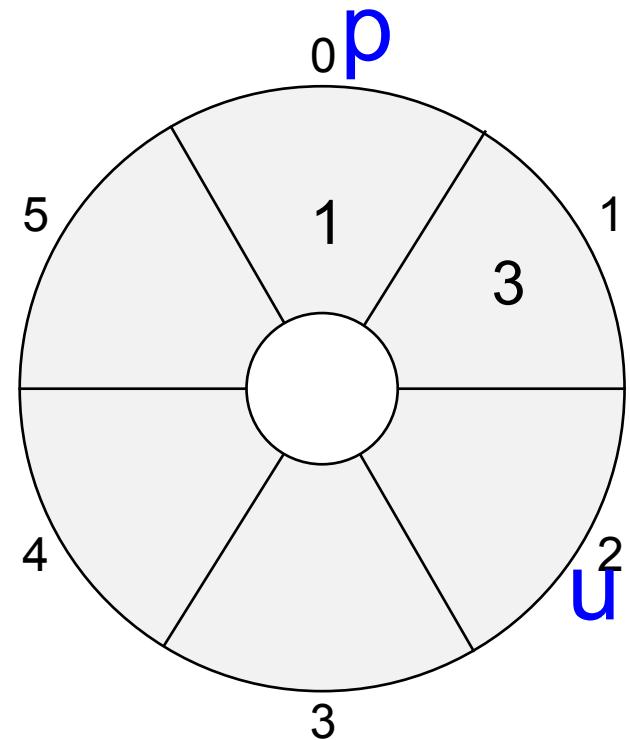
Algoritmo em Java

```
//Inserir(5)

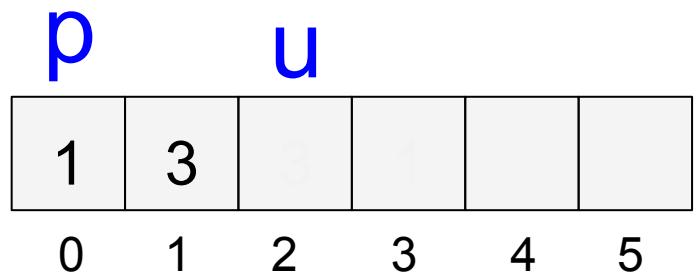
void inserir(int x) throws Exception {

    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



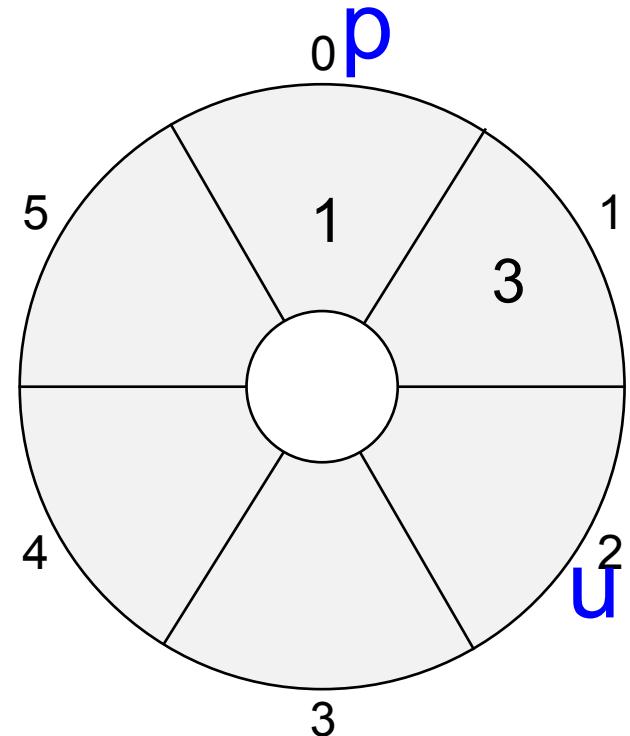
Algoritmo em Java

```
//Inserir(5)

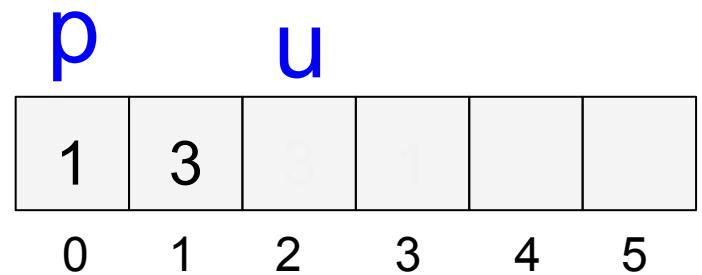
void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}

false: 2 + 1 % 6 == 0
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

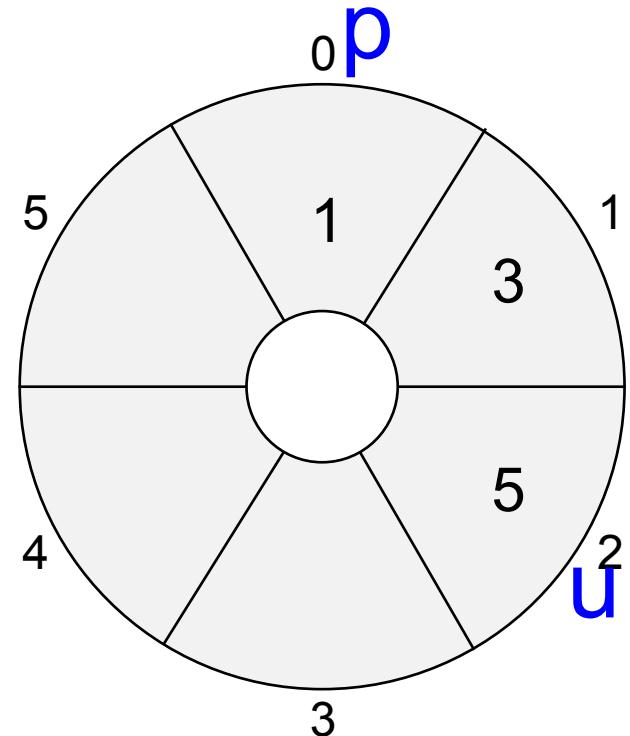


Algoritmo em Java

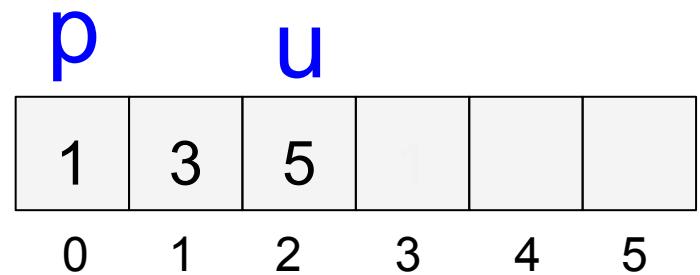
```
//Inserir(5)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

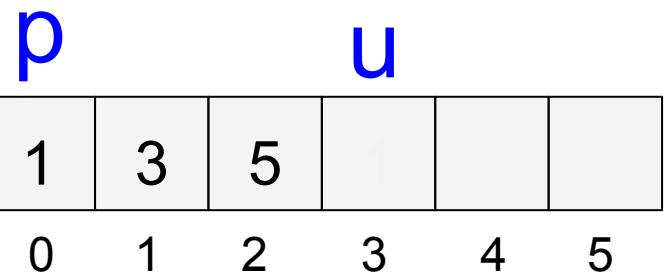
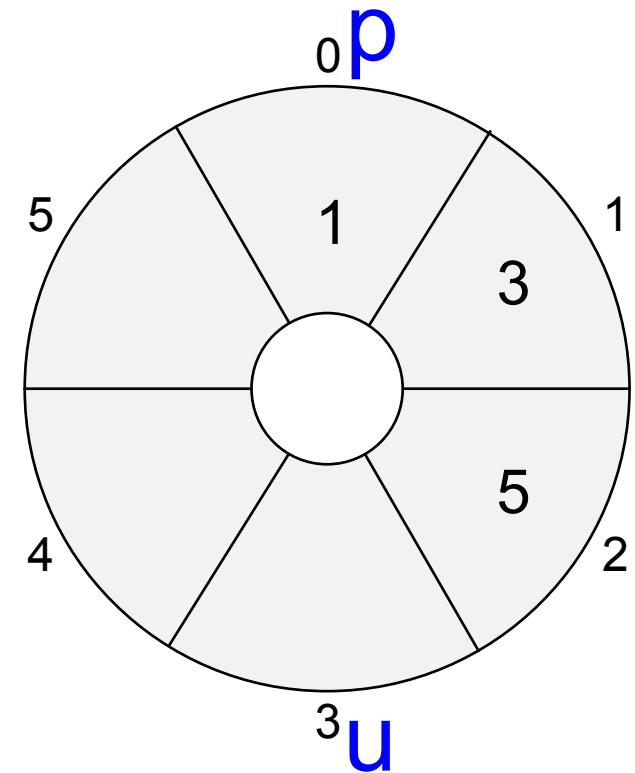


Algoritmo em Java

```
//Inserir(5)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



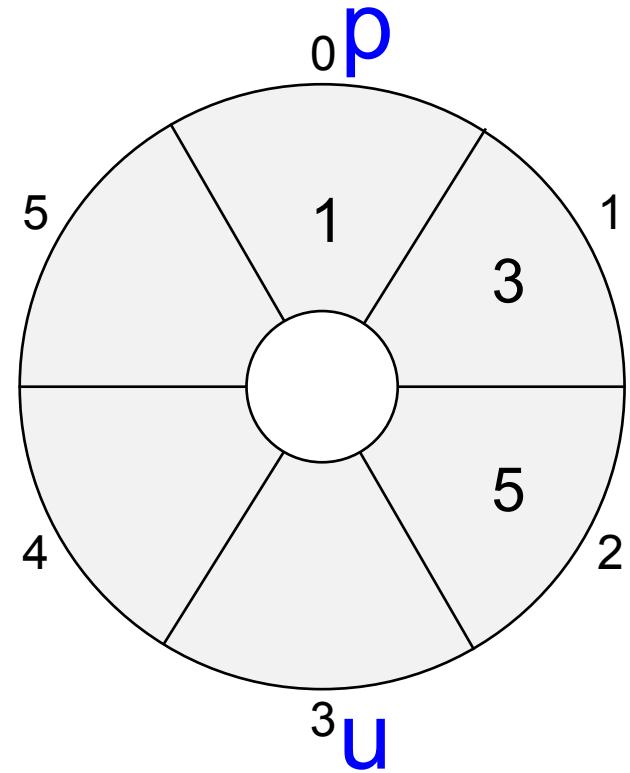
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

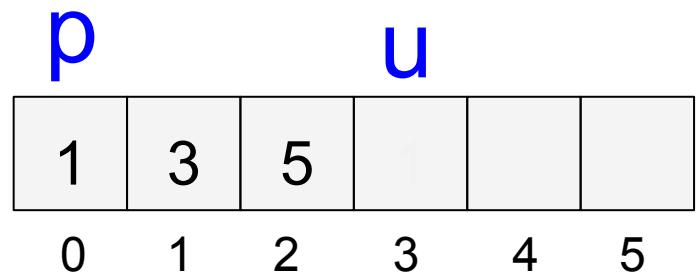
```
//Inserir(5)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



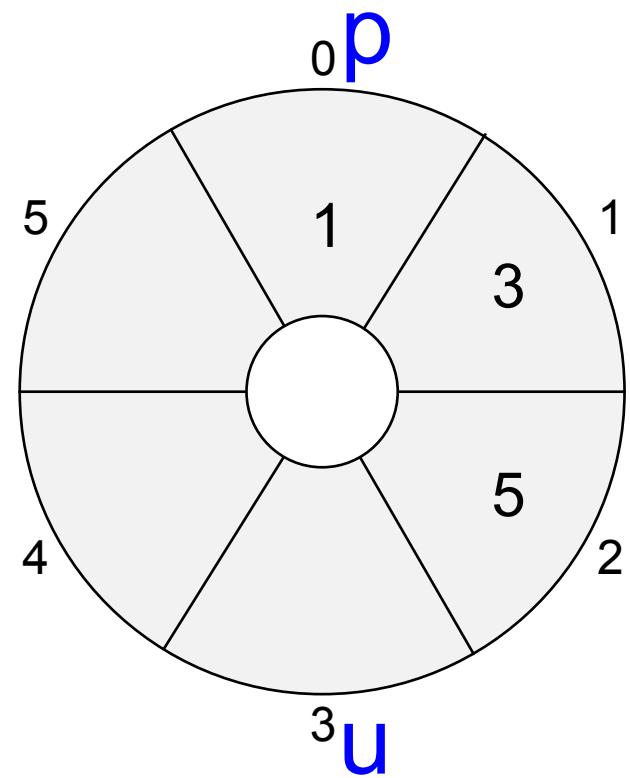
Algoritmo em Java

```
//Inserir(7)

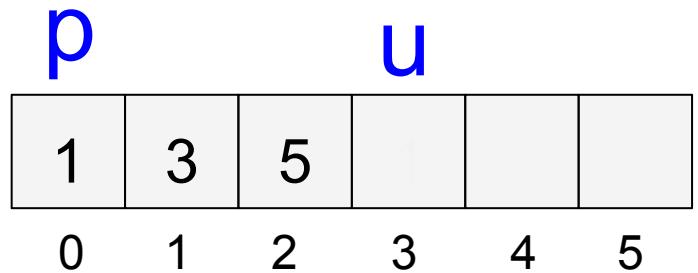
void inserir(int x) throws Exception {

    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



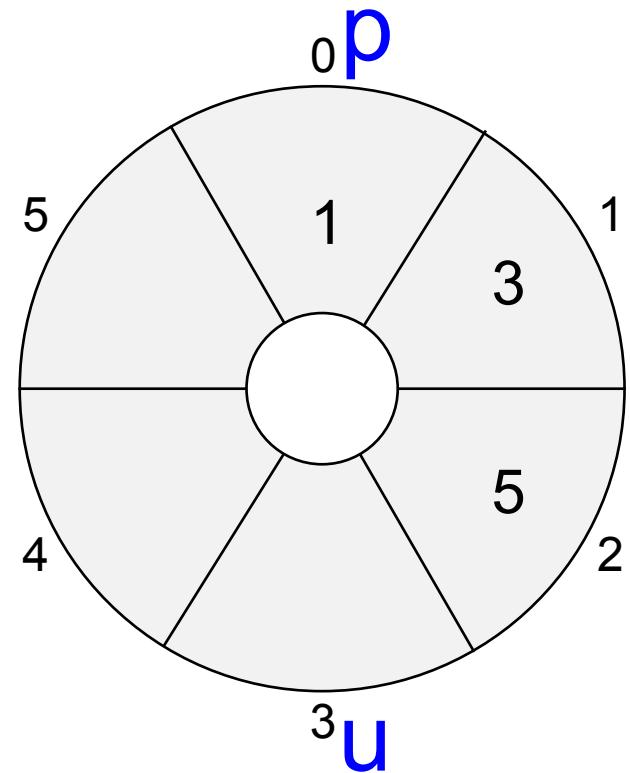
Algoritmo em Java

```
//Inserir(7)

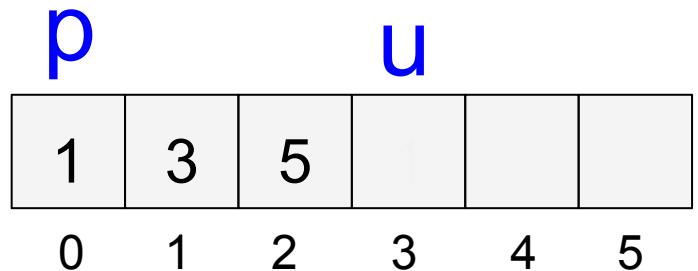
void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}

false: 3 + 1 % 6 == 0
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

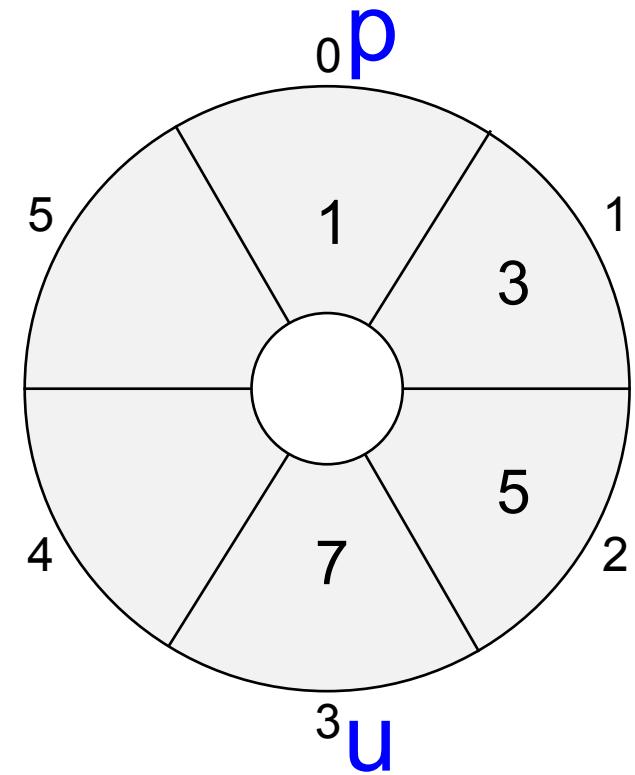


Algoritmo em Java

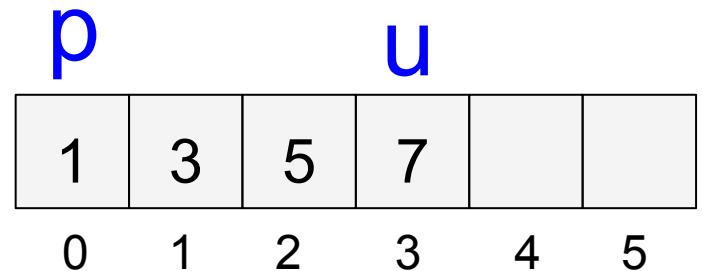
```
//Inserir(7)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

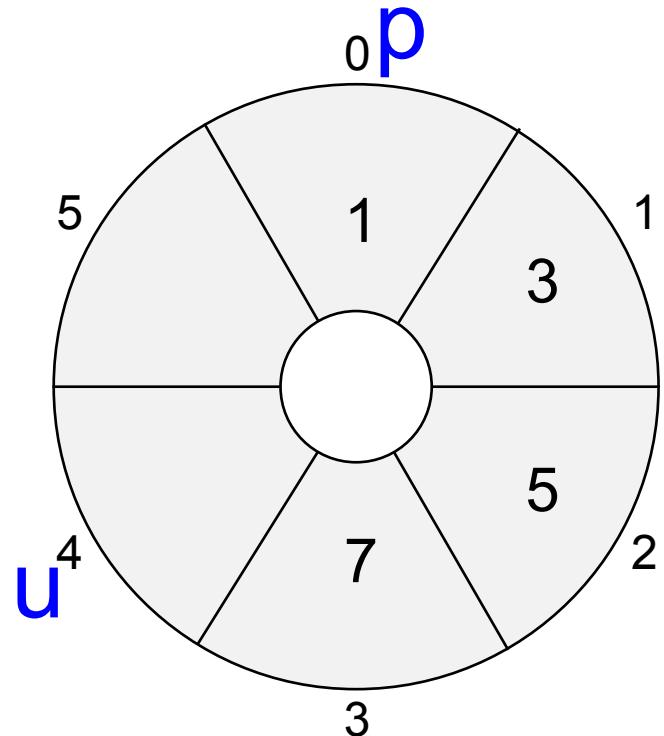


Algoritmo em Java

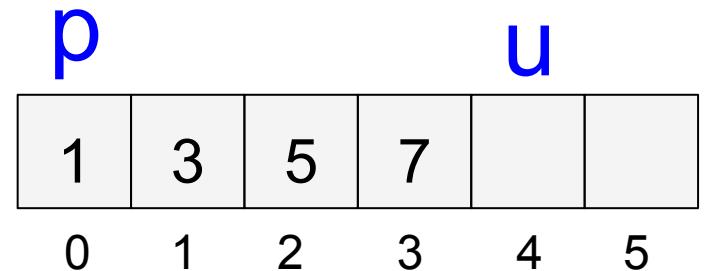
```
//Inserir(7)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

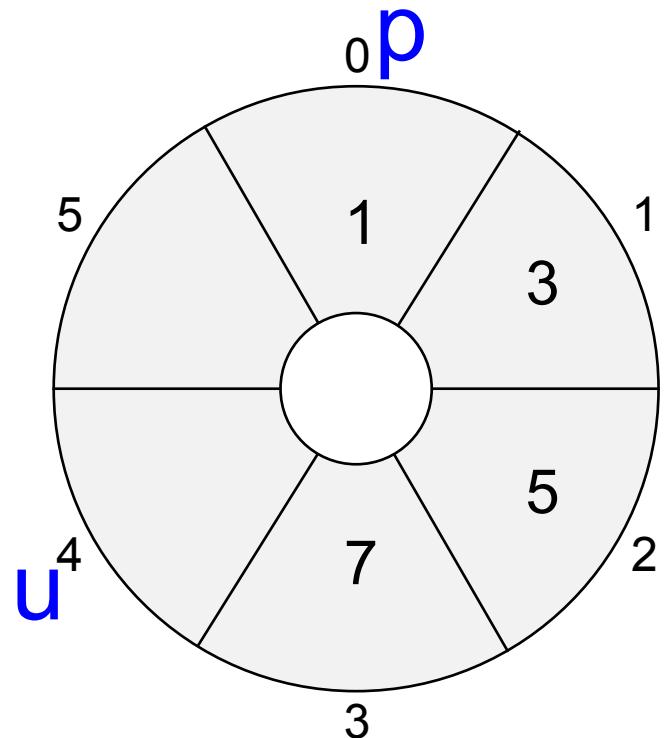


Algoritmo em Java

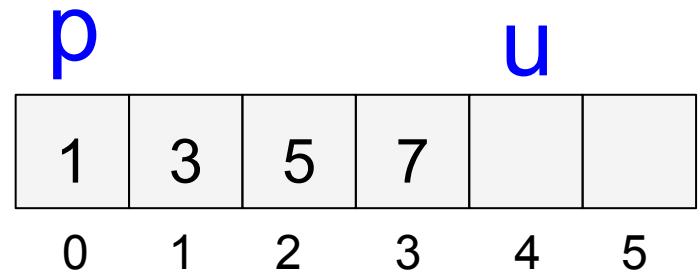
```
//Inserir(7)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



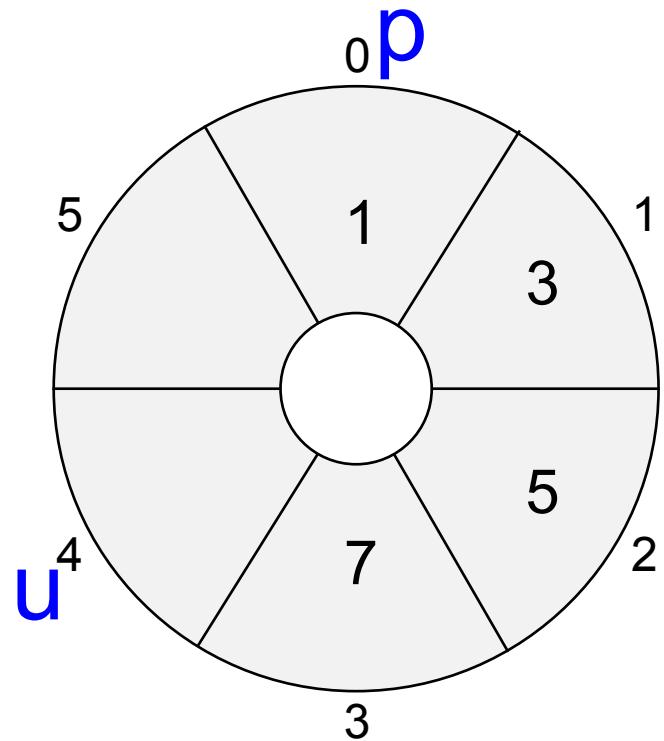
Algoritmo em Java

```
//Inserir(9)

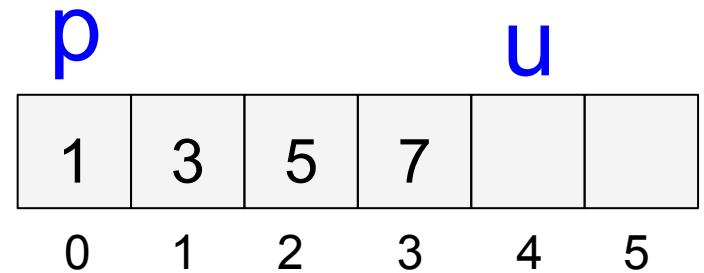
void inserir(int x) throws Exception {

    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



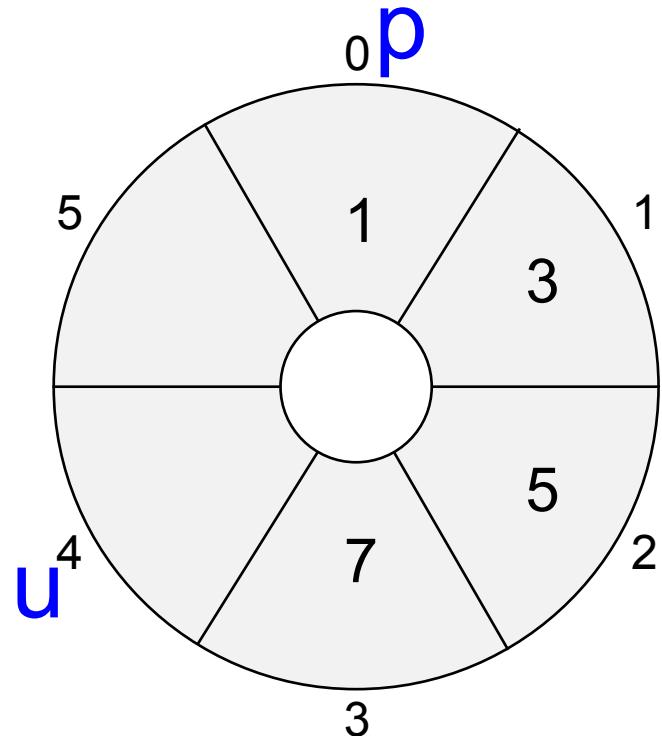
Algoritmo em Java

```
//Inserir(9)

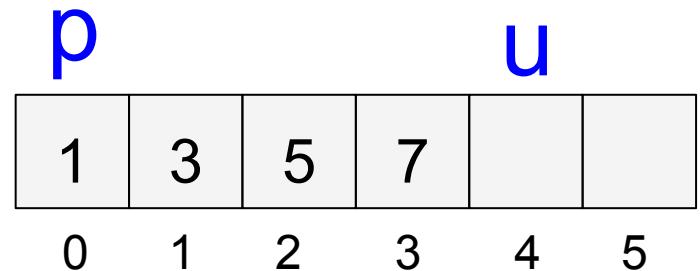
void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}

false: 4 + 1 % 6 == 0
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

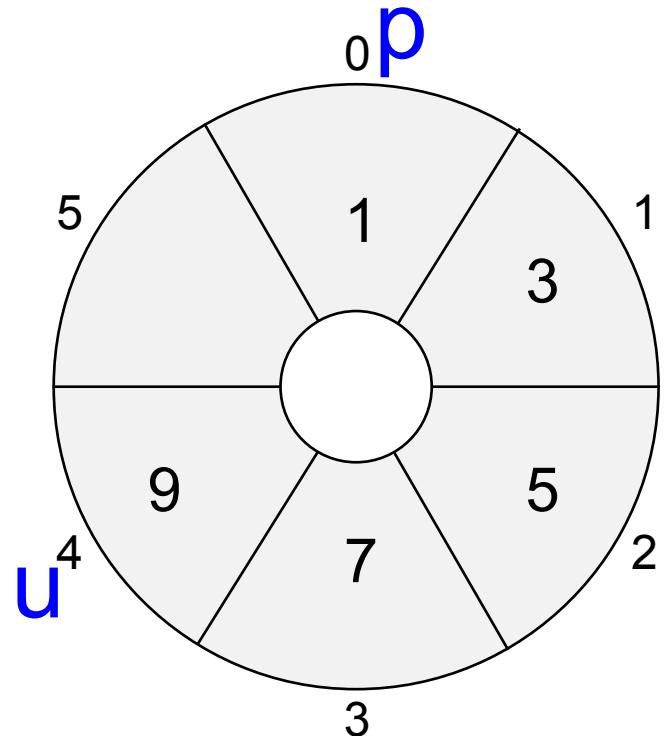


Algoritmo em Java

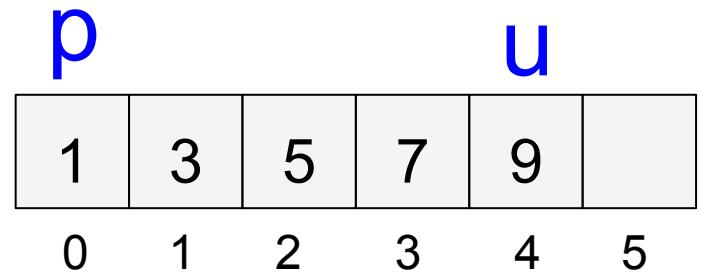
```
//Inserir(9)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

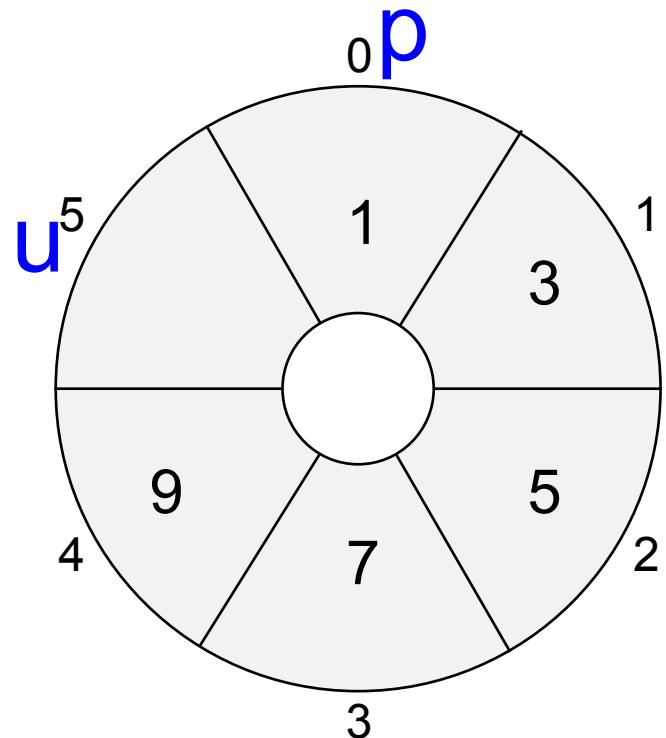


Algoritmo em Java

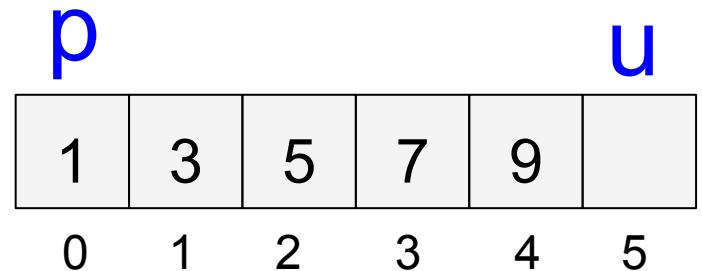
```
//Inserir(9)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

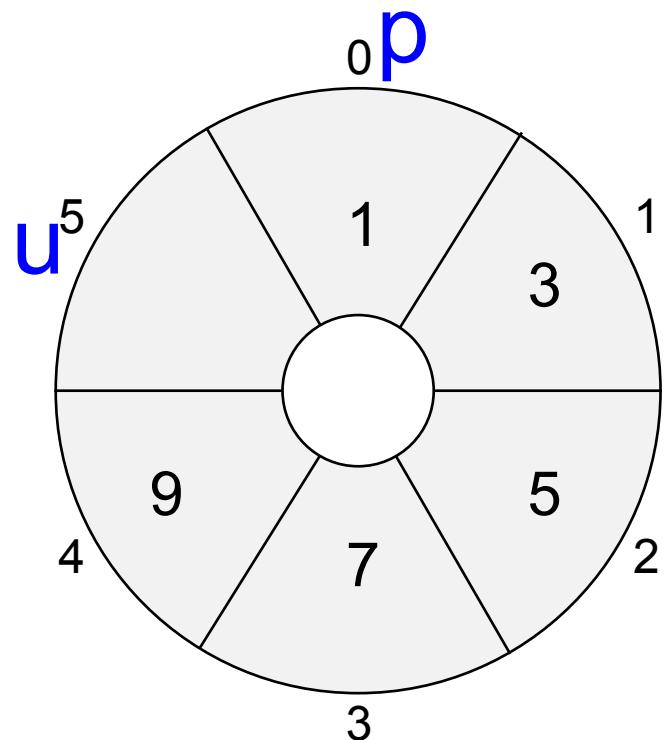


Algoritmo em Java

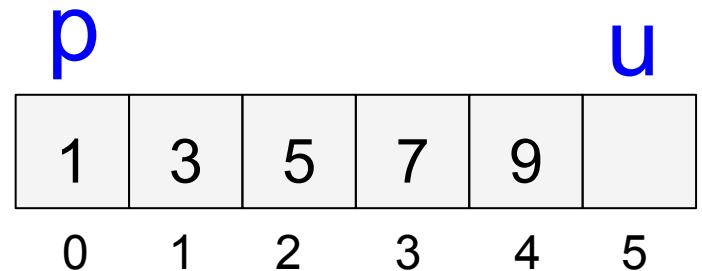
```
//Inserir(9)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



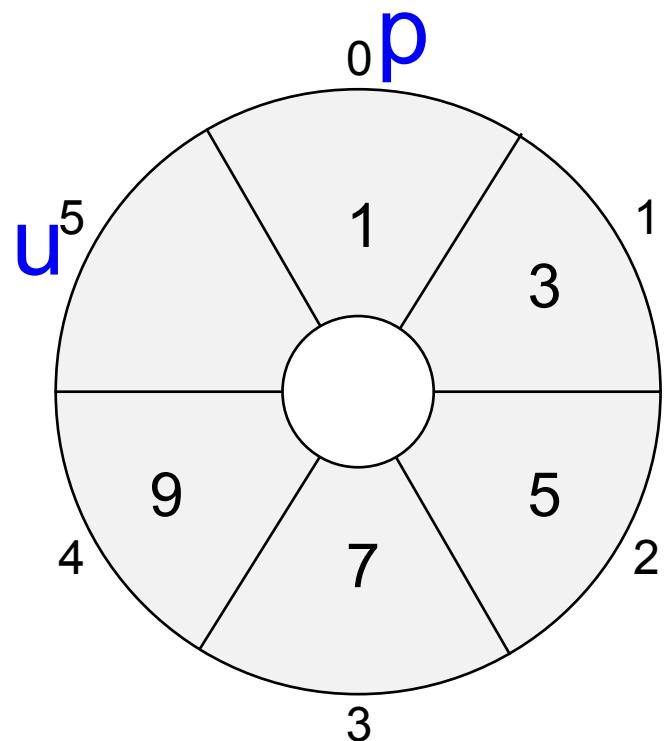
Algoritmo em Java

```
//Inserir(2)

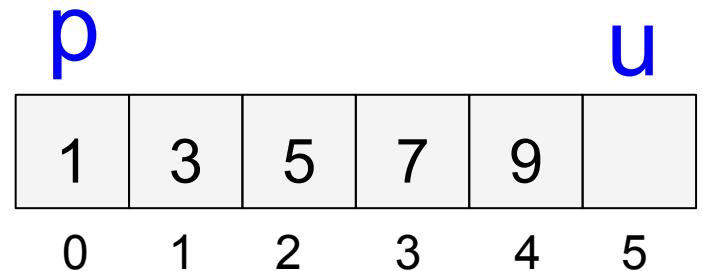
void inserir(int x) throws Exception {

    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

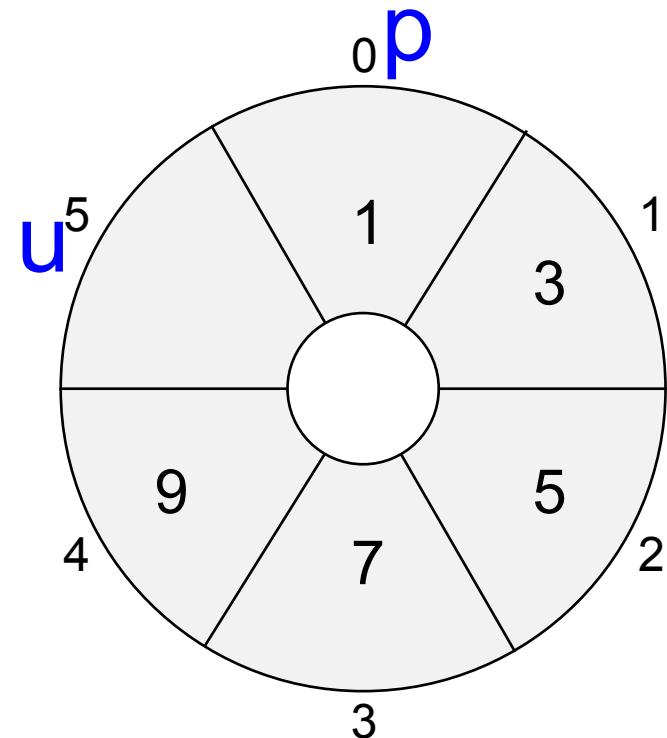


Algoritmo em Java

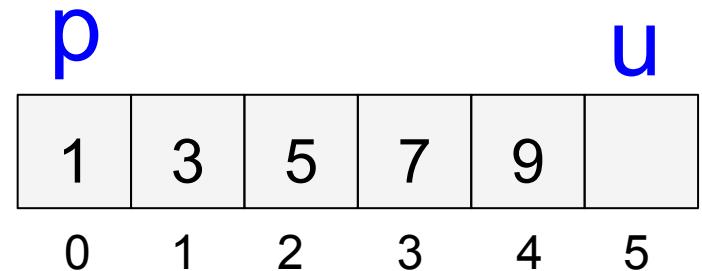
```
//Inserir(2)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");
    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```

true: $5 + 1 \% 6 == 0$



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

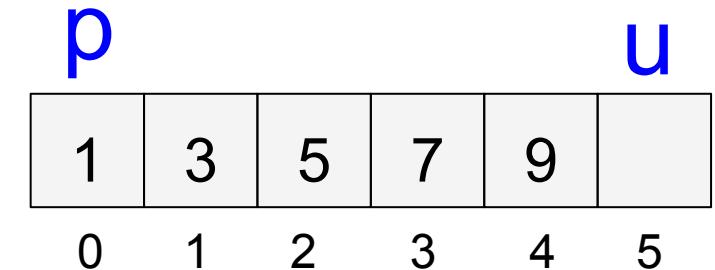
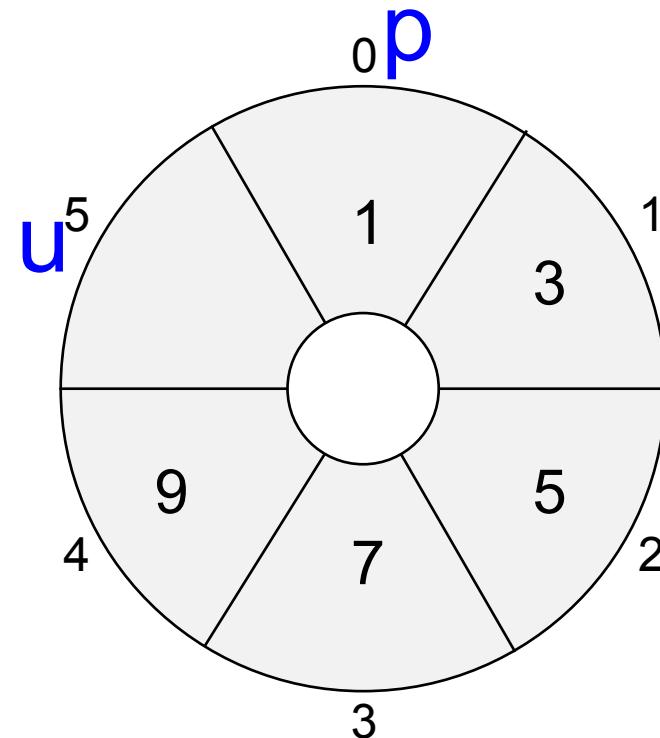
```
//Inserir(2)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");
    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```

true: $5 + 1 \% 6 == 0$

Como nossa fila tem tamanho cinco, não conseguimos alocar mais elementos

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

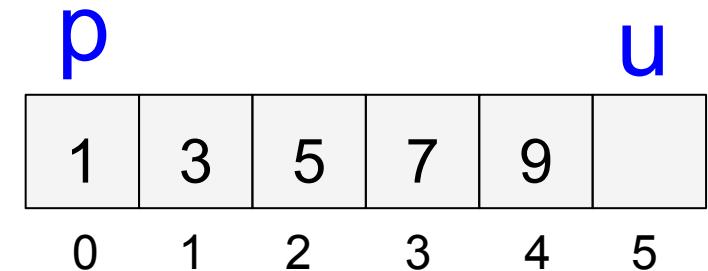
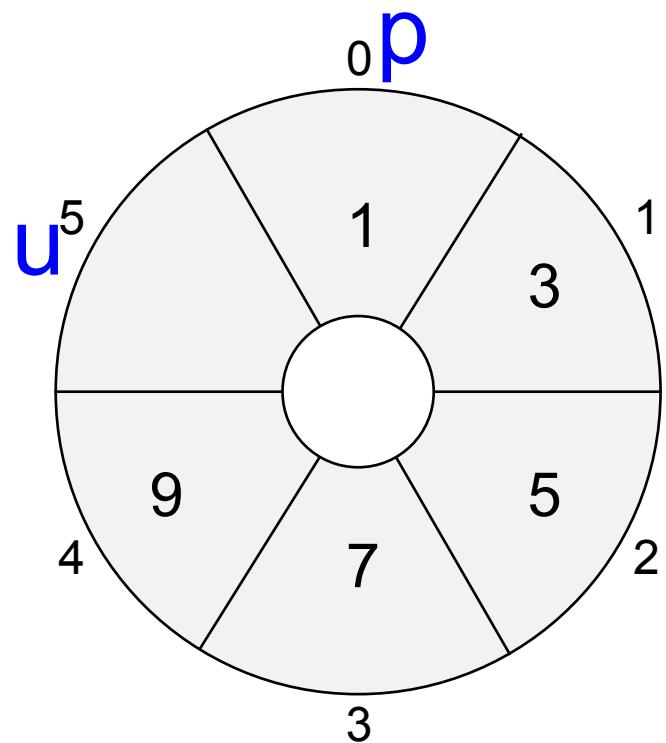
```
//Inserir(2)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");
    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```

true: $5 + 1 \% 6 == 0$

Vamos para a próxima operação, um remover

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



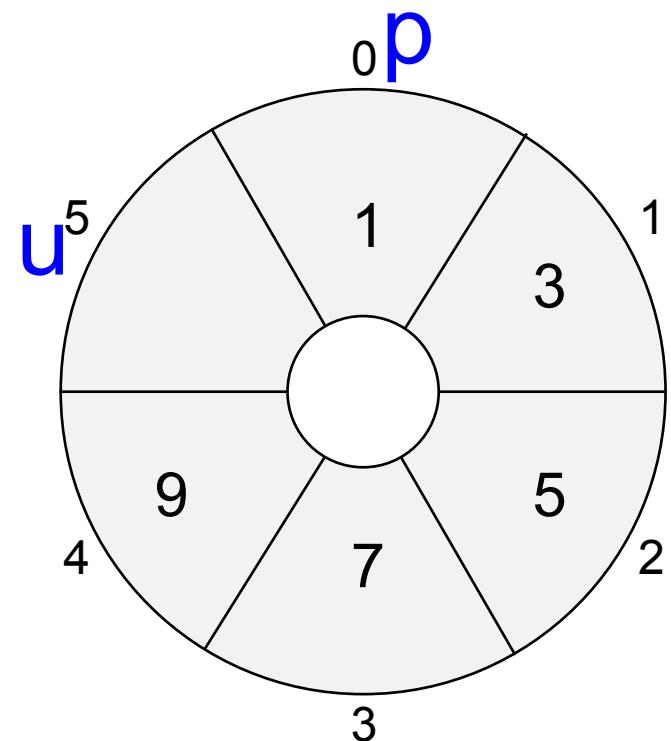
Algoritmo em Java

```
//Remover()

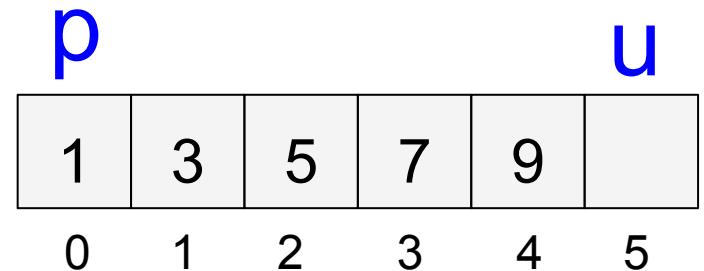
int remover() throws Exception {

    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



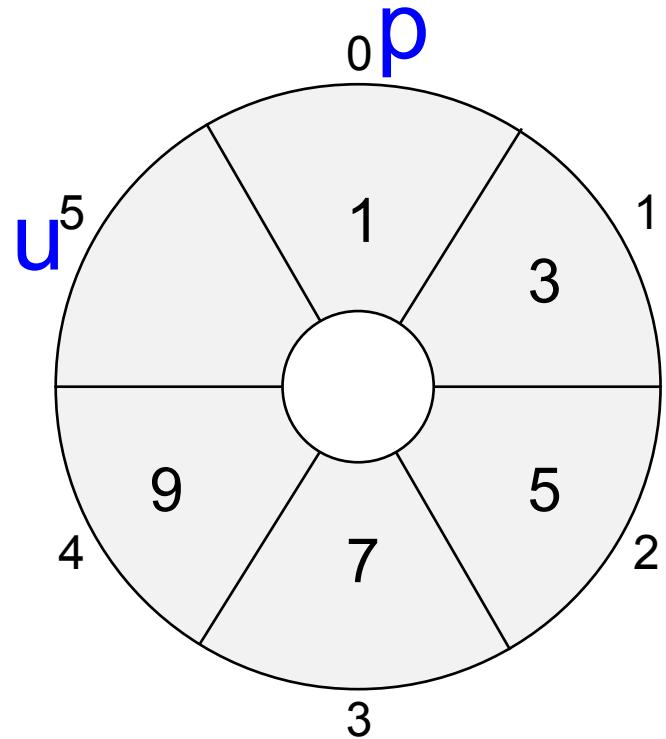
Algoritmo em Java

```
//Remover()

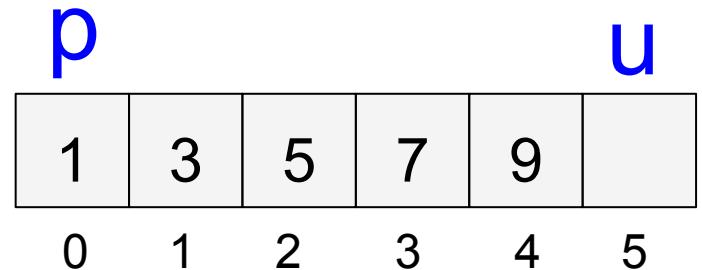
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}

false: 0 == 5
```

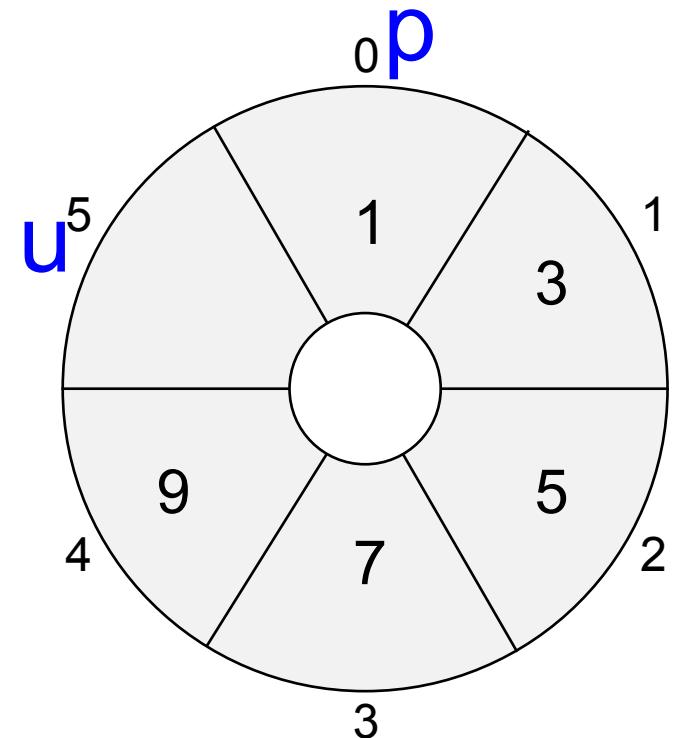


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

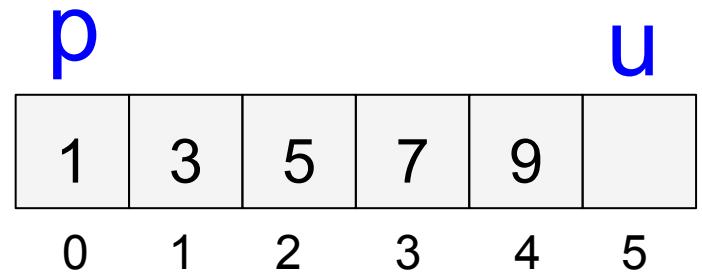


Algoritmo em Java

```
//Remover()  
  
int remover() throws Exception {  
  
    if (primeiro == ultimo)  
        throw new Exception("Erro!");  
  
    int resp = array[primeiro];  
    primeiro = (primeiro + 1) % array.length;  
    return resp;  
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações $I(1)$, $I(3)$, $I(5)$, $I(7)$, $I(9)$, $I(2)$, $R()$, $R()$, $I(4)$, $I(6)$, $R()$, $I(8)$, $M()$



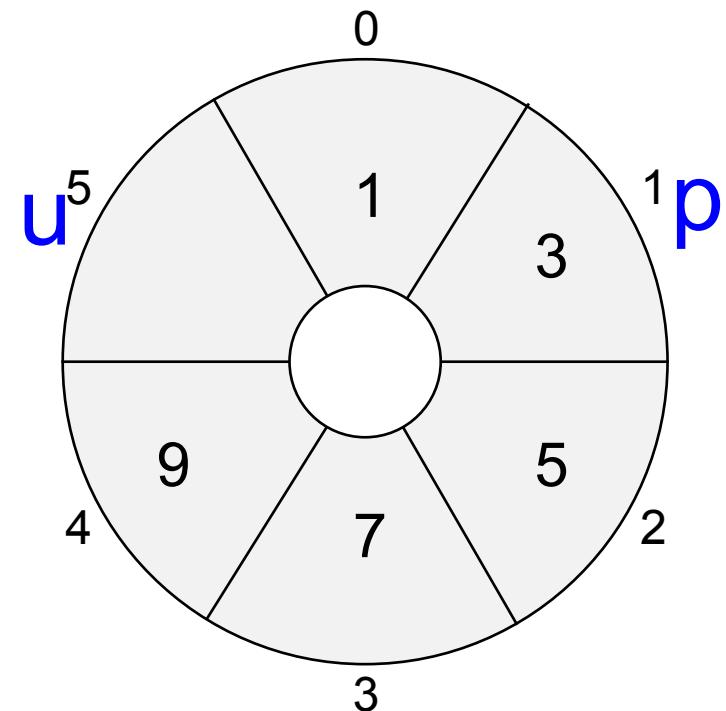
Algoritmo em Java

```
//Remover()

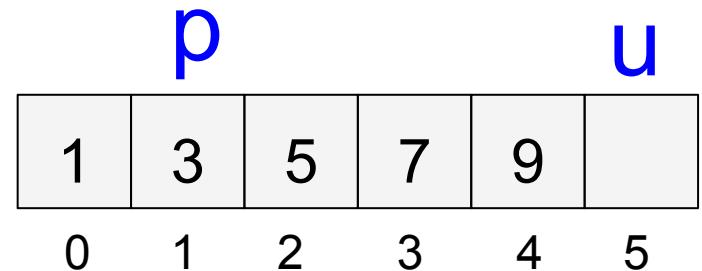
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 1



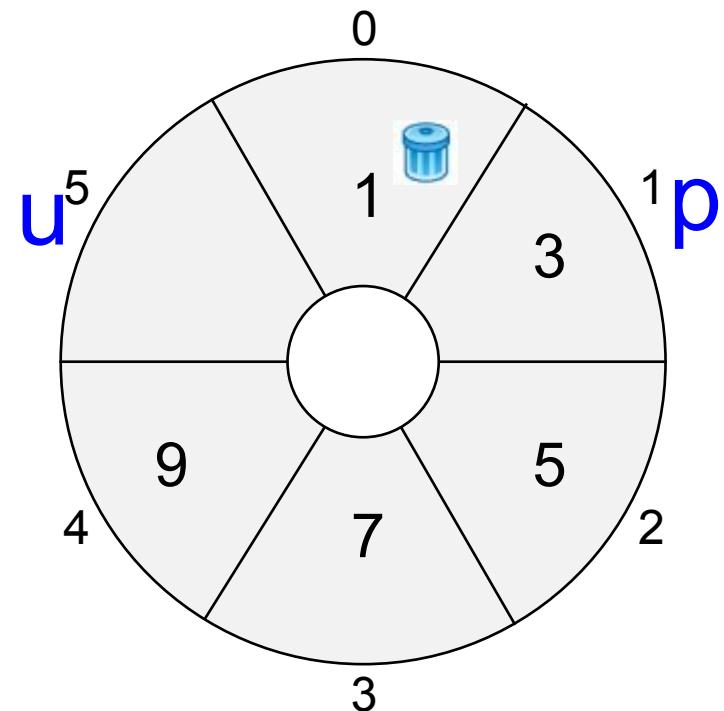
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



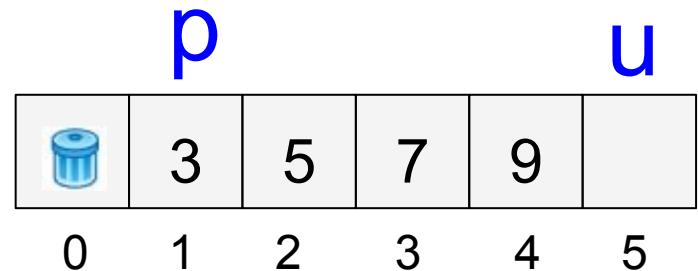
Algoritmo em Java

```
//Remover()
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");
    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 1



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

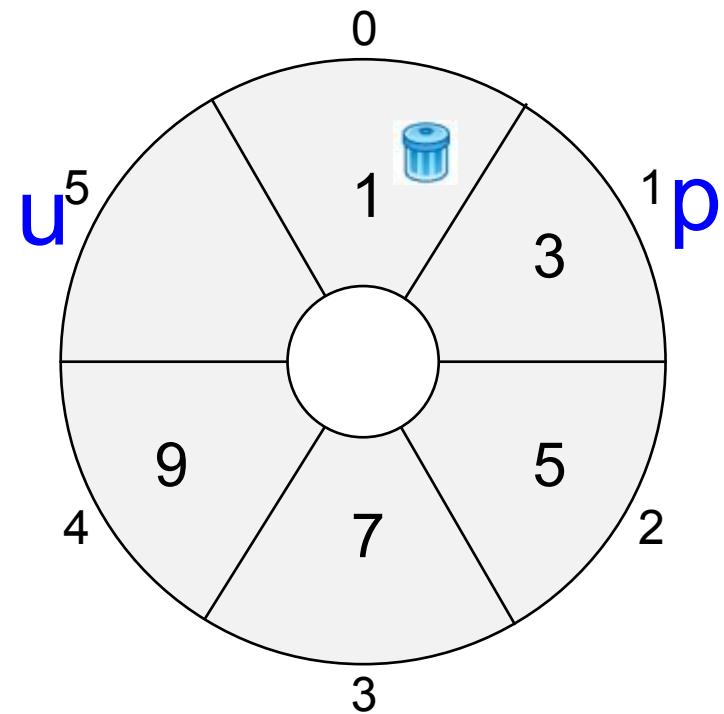


Algoritmo em Java

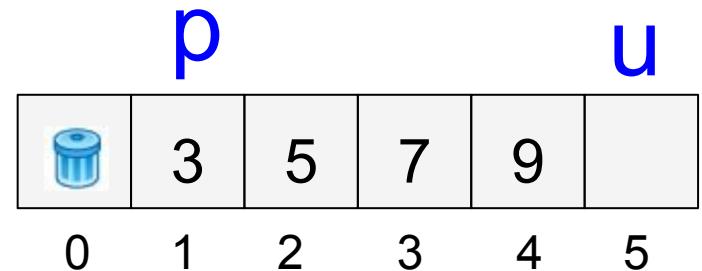
```
//Remover()

int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



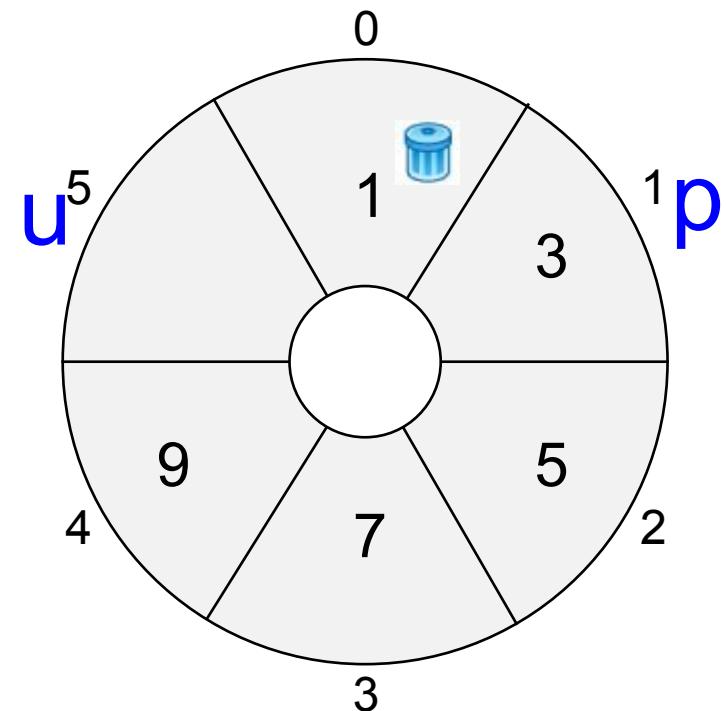
Algoritmo em Java

```
//Remover()

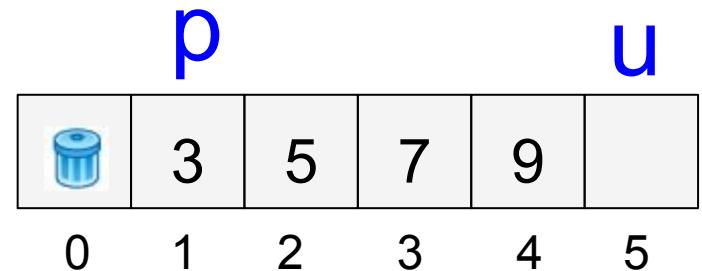
int remover() throws Exception {

    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



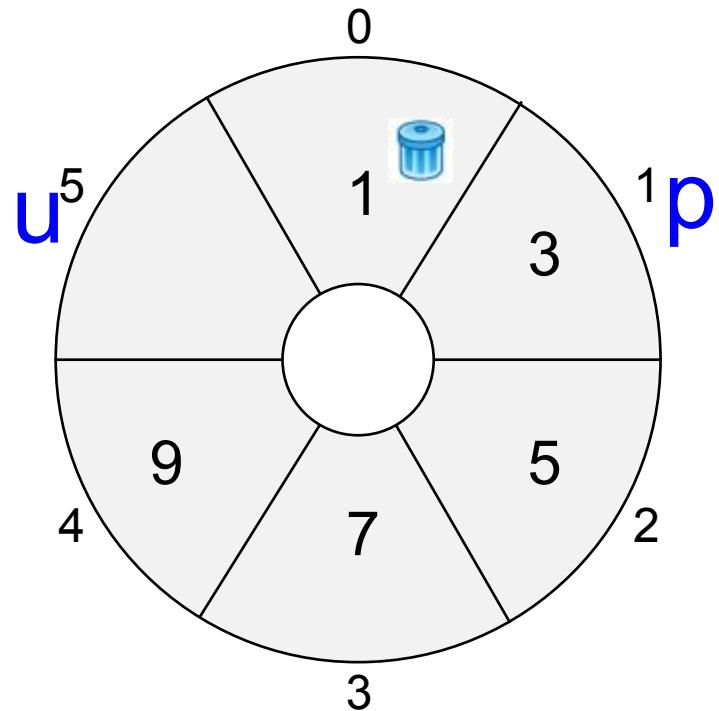
Algoritmo em Java

```
//Remover()

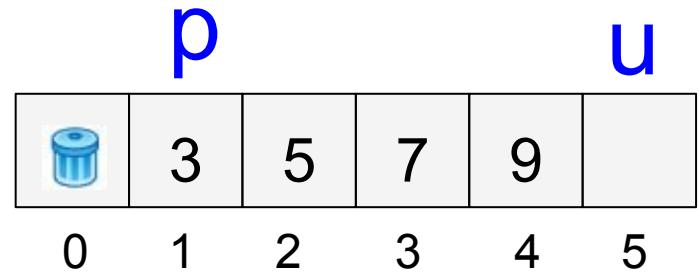
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}

false: 1 == 5
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



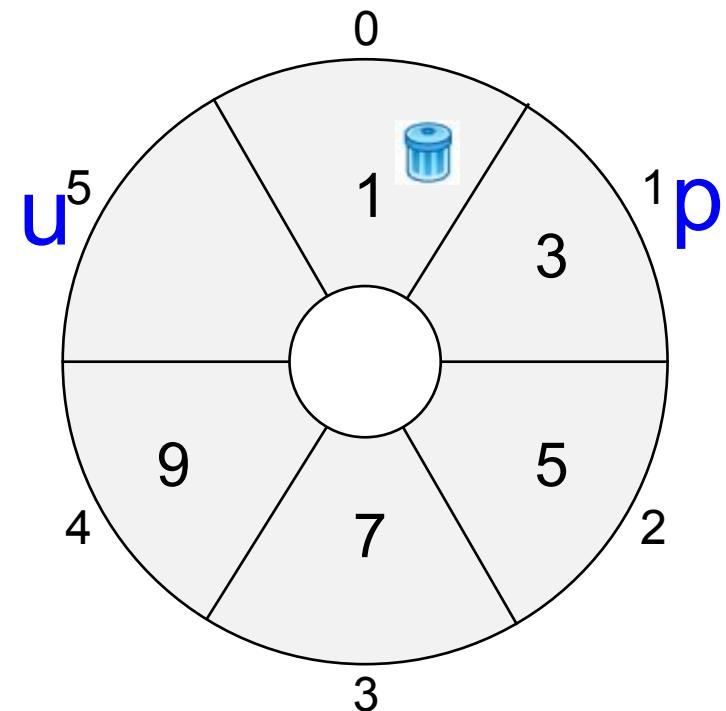
Algoritmo em Java

```
//Remover()

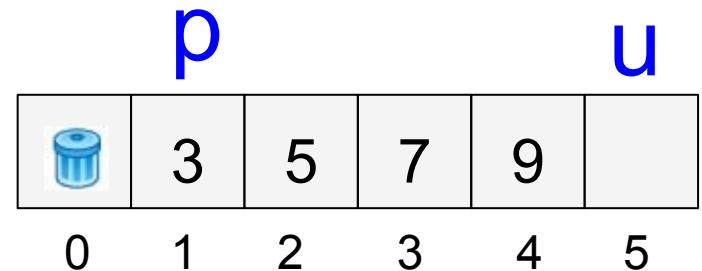
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 3



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



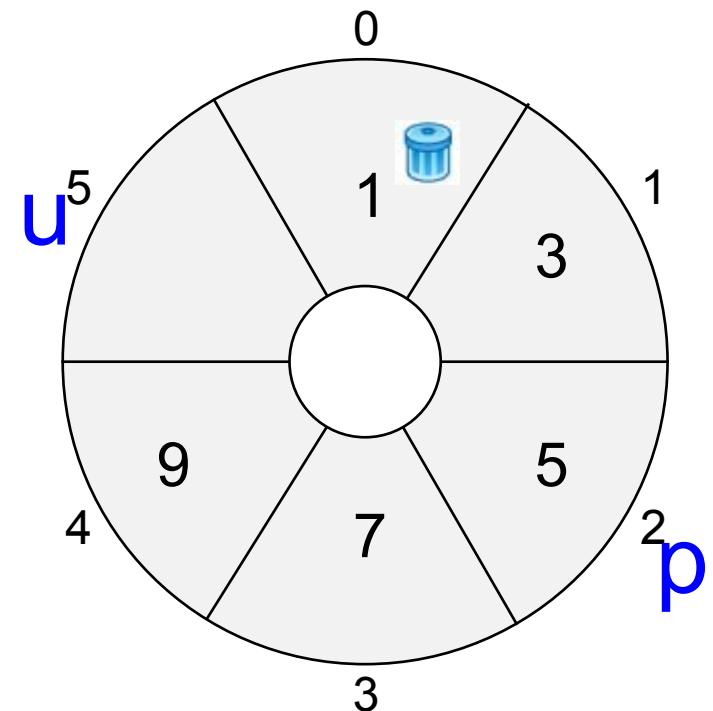
Algoritmo em Java

```
//Remover()

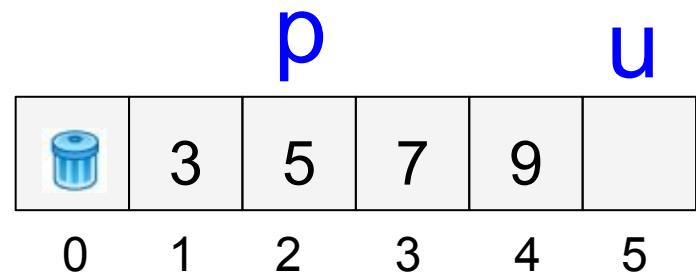
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 3



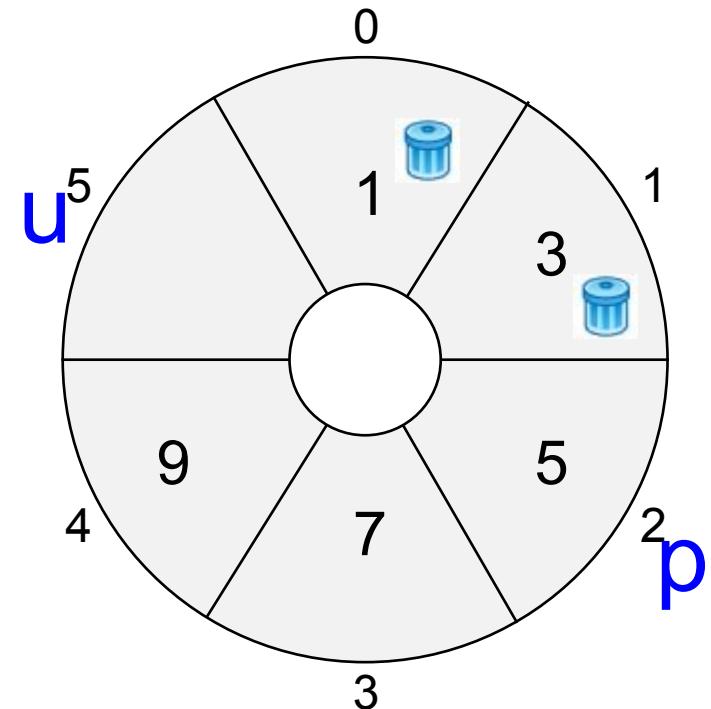
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



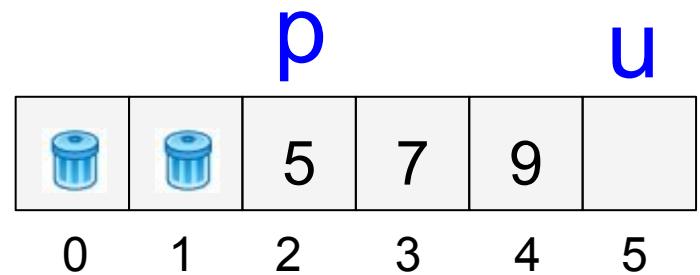
Algoritmo em Java

```
//Remover()
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");
    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 3



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

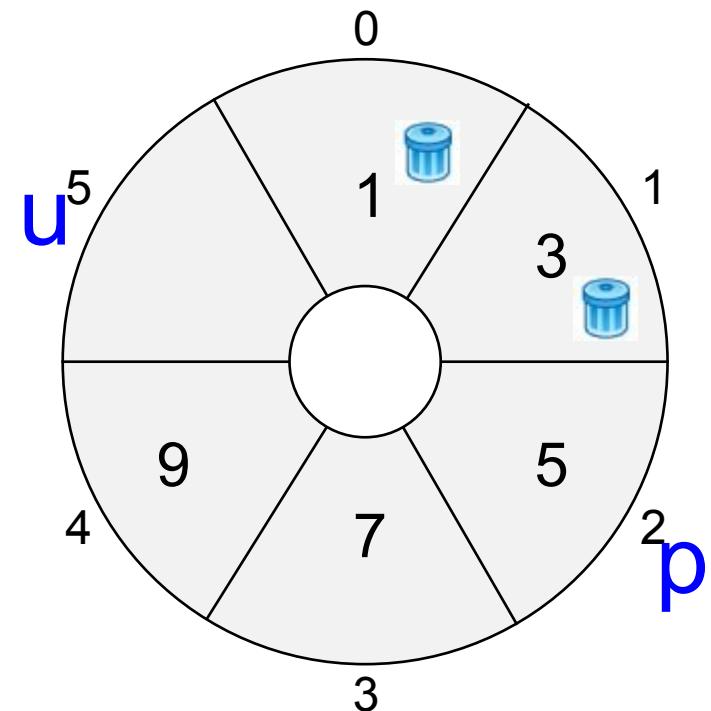


Algoritmo em Java

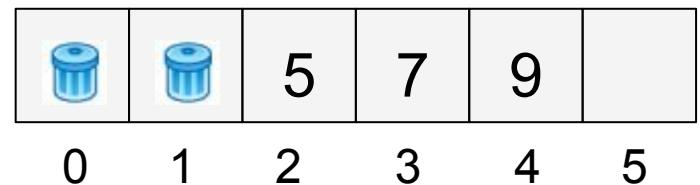
```
//Remover()

int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```



p u



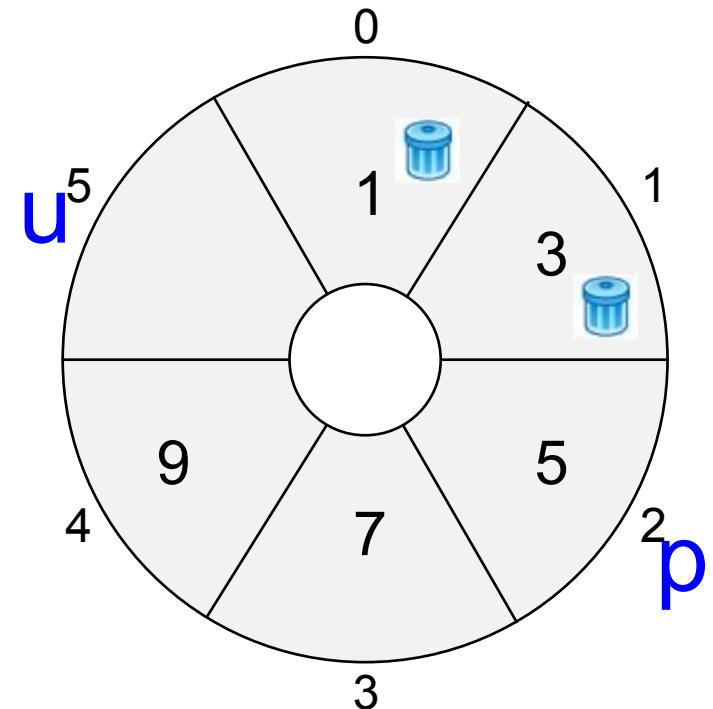
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

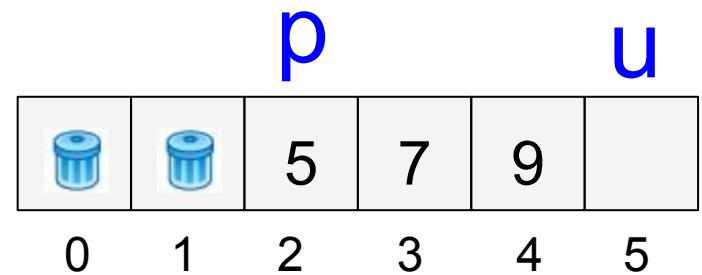
```
//Inserir(4)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



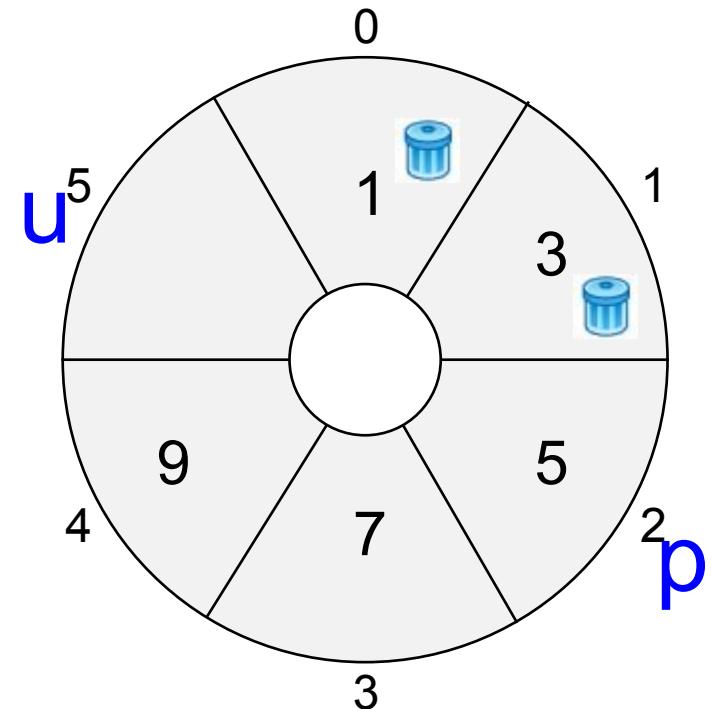
Algoritmo em Java

```
//Inserir(4)

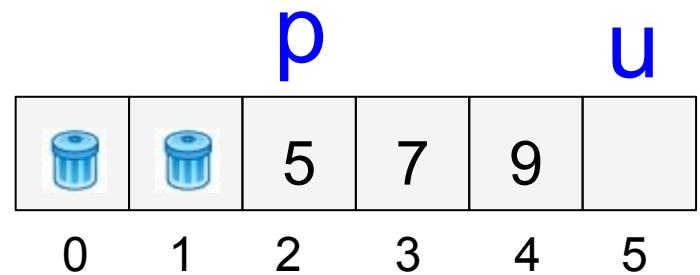
void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}

false: 5 + 1 % 6 == 2
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

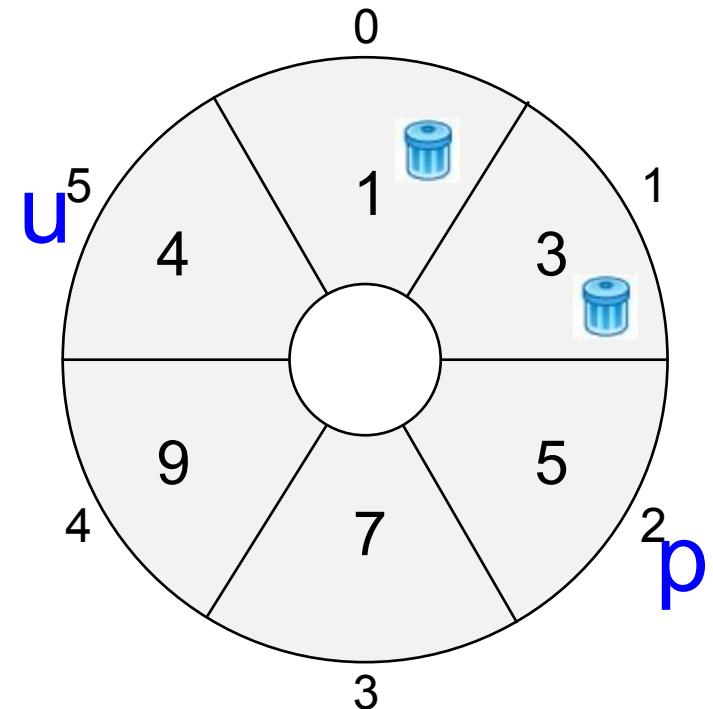


Algoritmo em Java

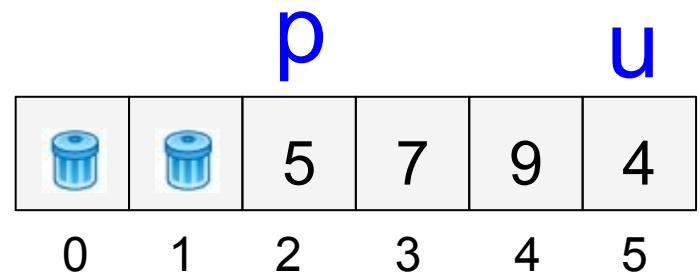
```
//Inserir(4)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

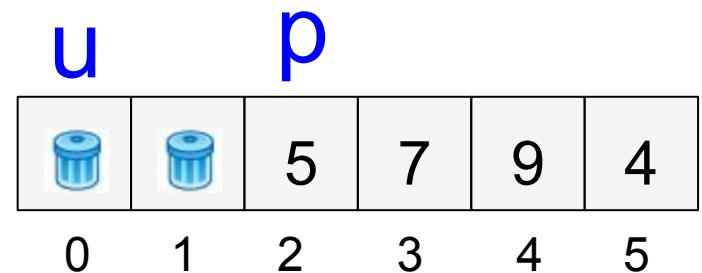
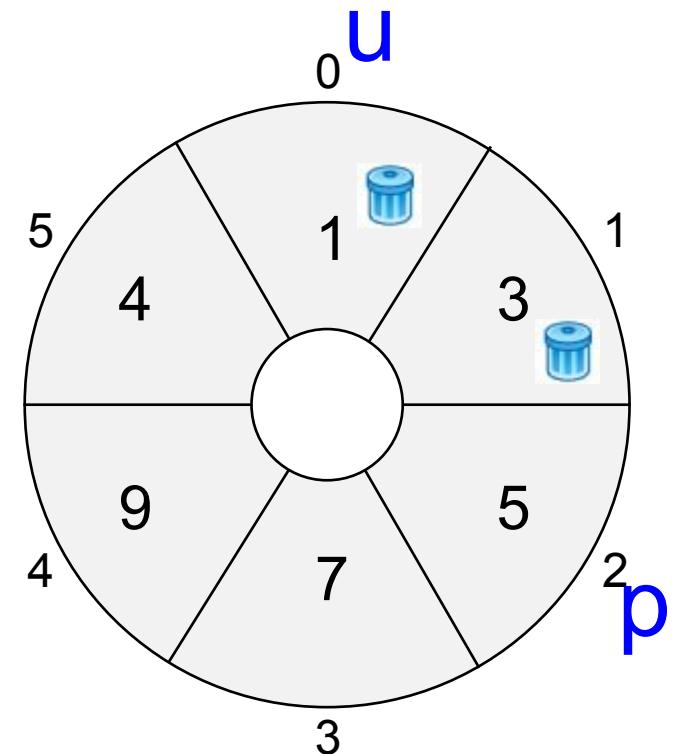


Algoritmo em Java

```
//Inserir(4)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



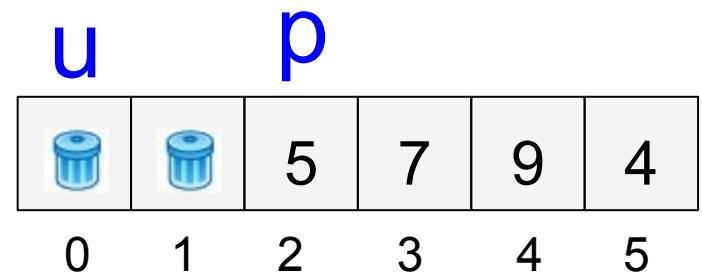
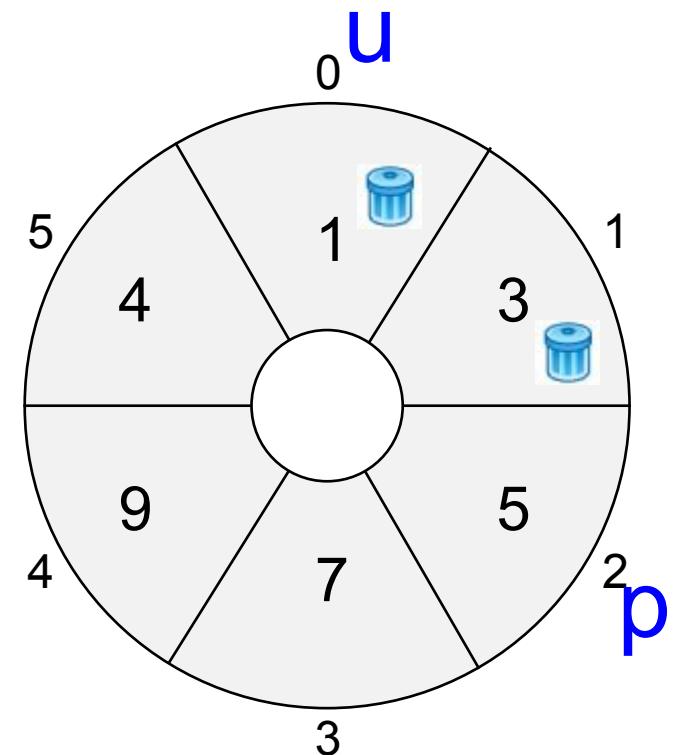
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em Java

```
//Inserir(4)

void inserir(int x) throws Exception {
    if ((ultimo + 1) % array.length == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), **I(4)**, I(6), R(), I(8), M()

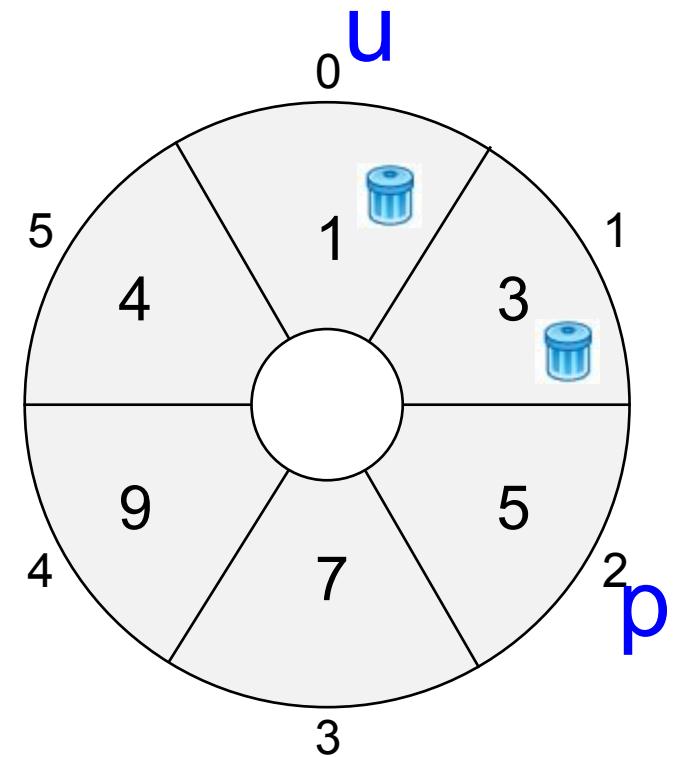
Algoritmo em Java

```
//Inserir(6)

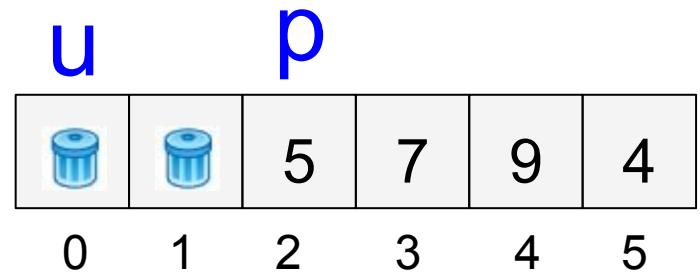
void inserir(int x) throws Exception {

    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), **I(6)**, R(), I(8), M()



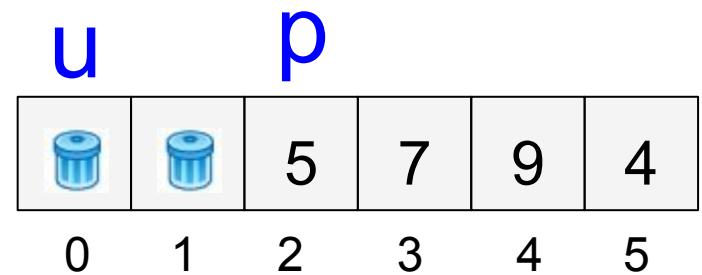
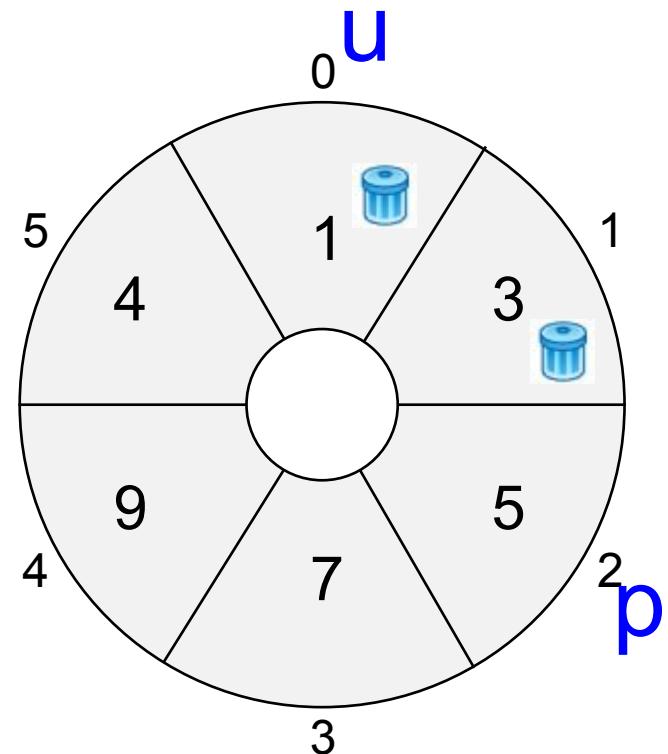
Algoritmo em Java

```
//Inserir(6)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}

false: 0 + 1 % 6 == 2
```



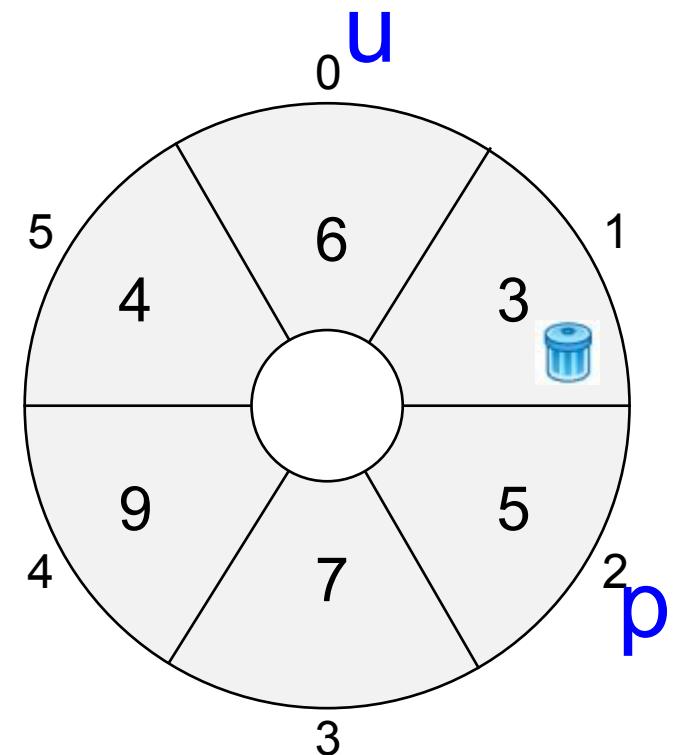
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), **I(6)**, R(), I(8), M()

Algoritmo em Java

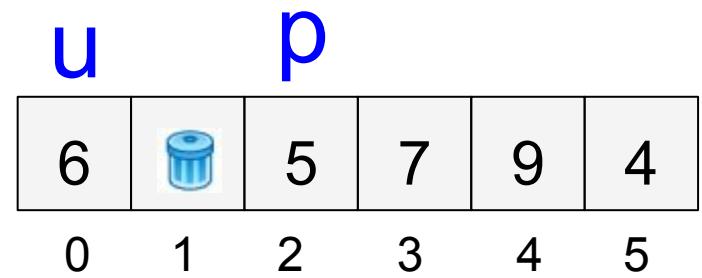
```
//Inserir(6)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), **I(6)**, R(), I(8), M()

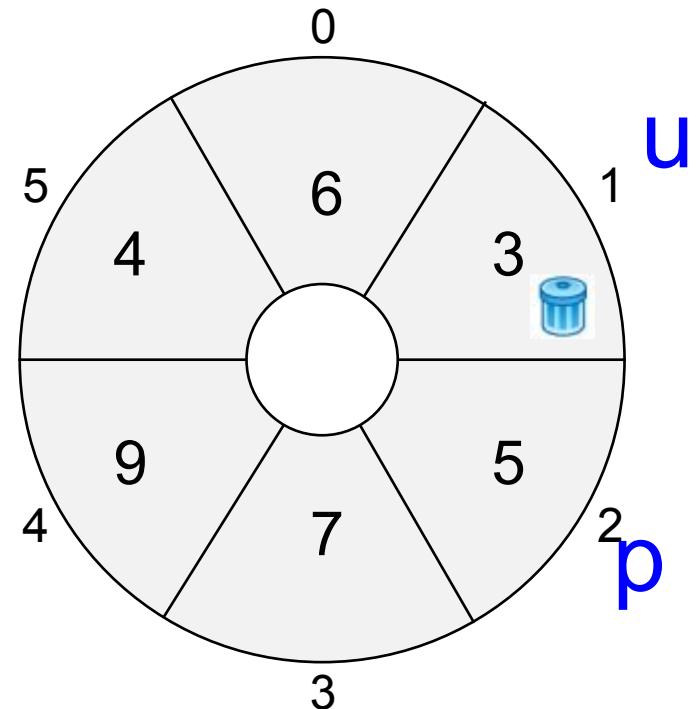


Algoritmo em Java

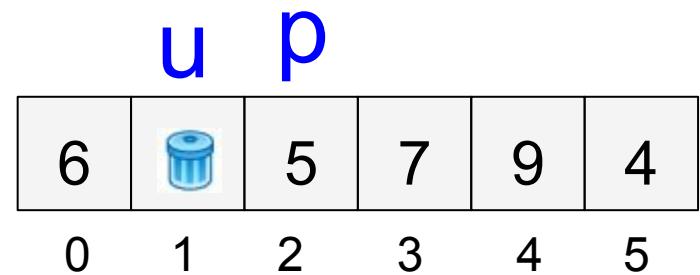
```
//Inserir(6)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), **I(6)**, R(), I(8), M()

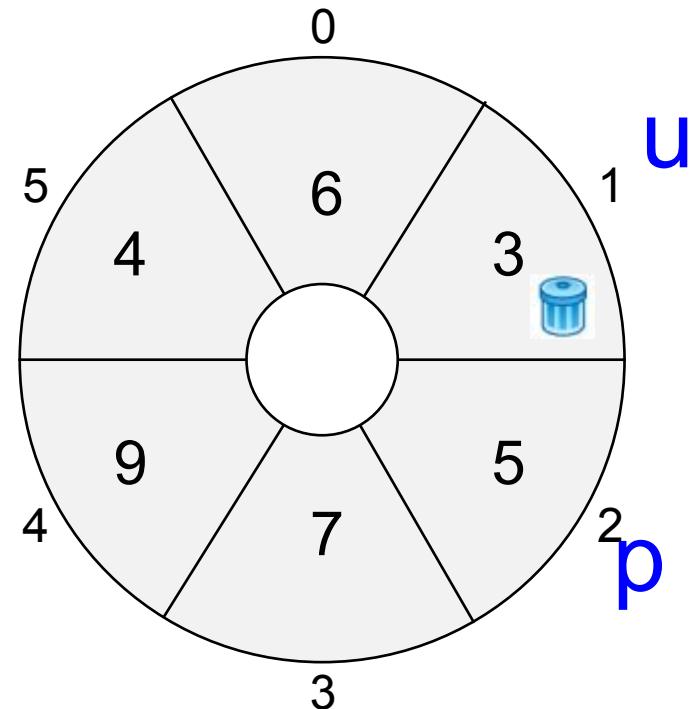


Algoritmo em Java

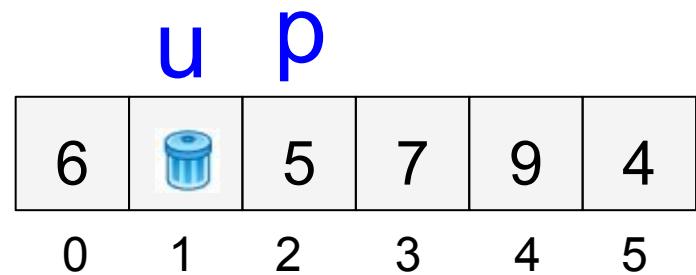
```
//Inserir(6)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), **I(6)**, R(), I(8), M()



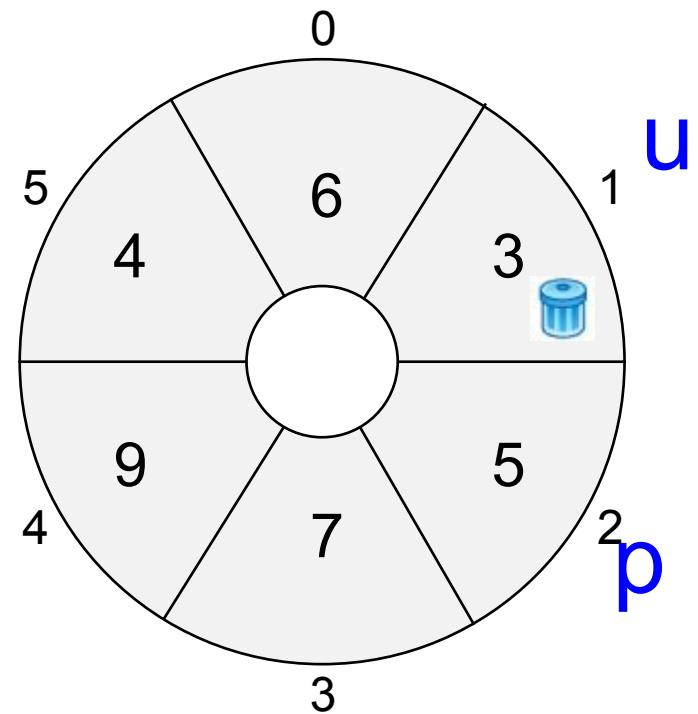
Algoritmo em Java

```
//Remover()

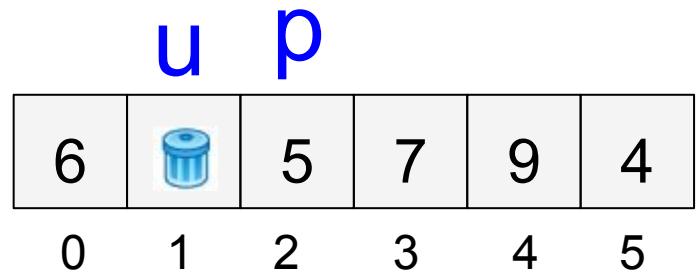
int remover() throws Exception {

    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

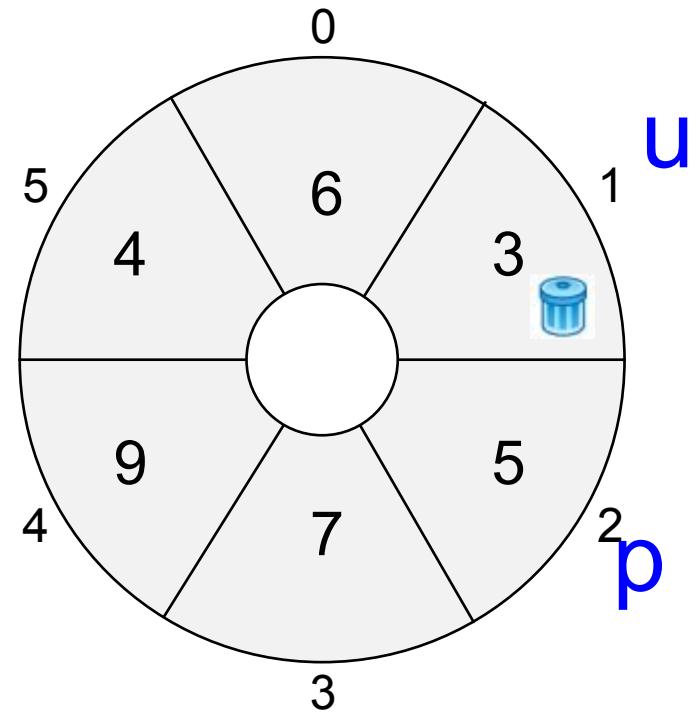


Algoritmo em Java

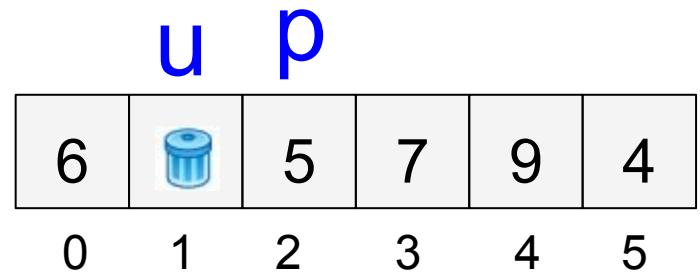
```
//Remover()

int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
false: 2 == 1
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



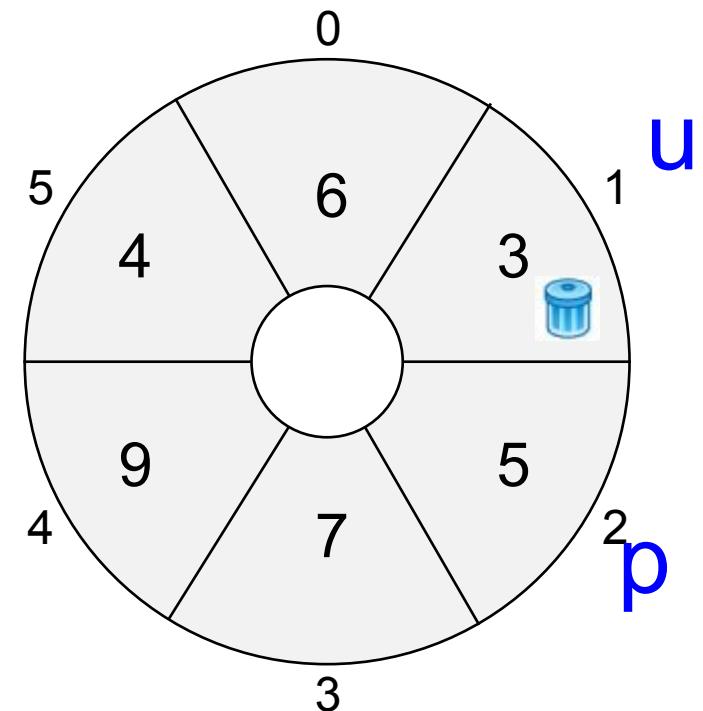
Algoritmo em Java

```
//Remover()

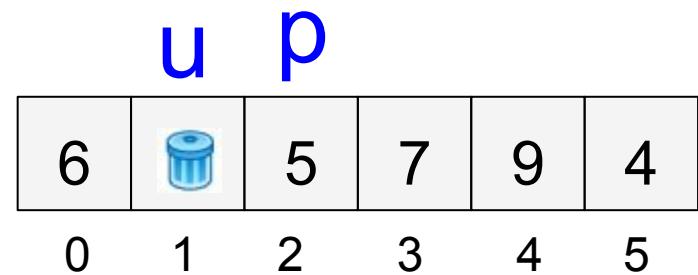
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 5



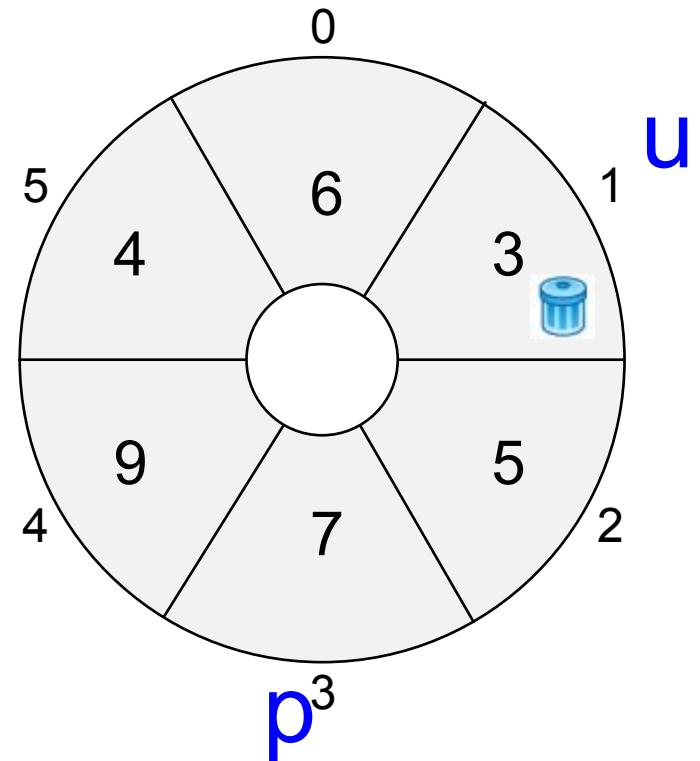
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



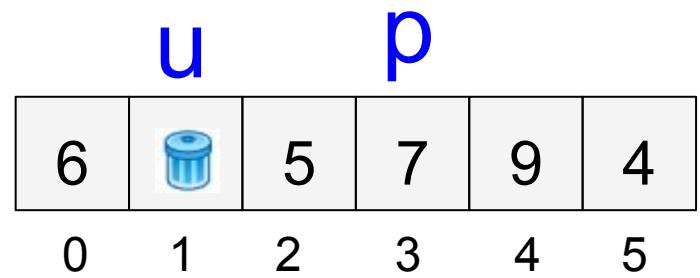
Algoritmo em Java

```
//Remover()
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");
    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 5



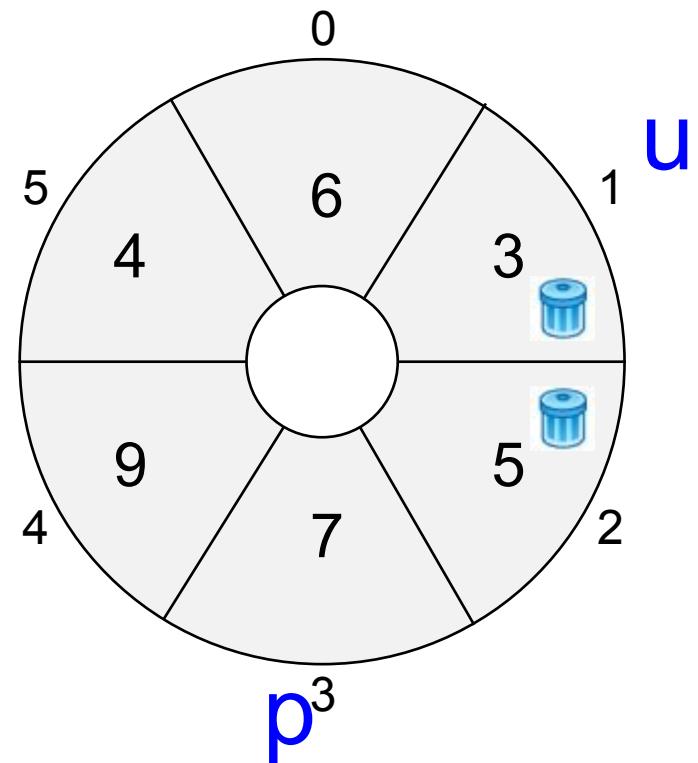
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



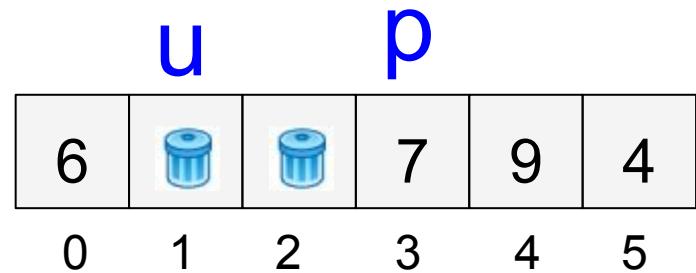
Algoritmo em Java

```
//Remover()
int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");
    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```

resp 5



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

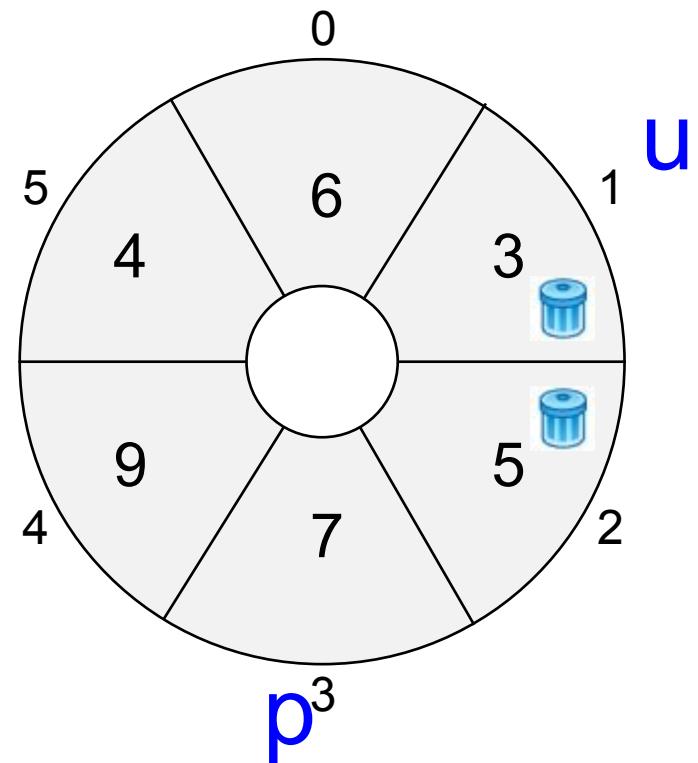


Algoritmo em Java

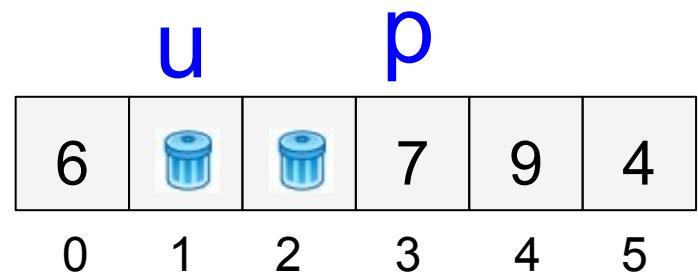
```
//Remover()

int remover() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % array.length;
    return resp;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

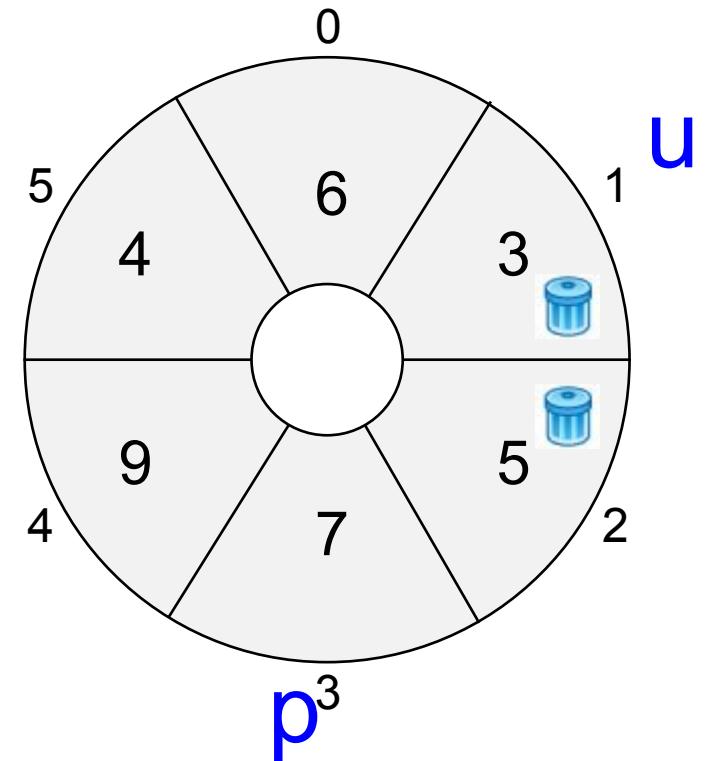


Algoritmo em Java

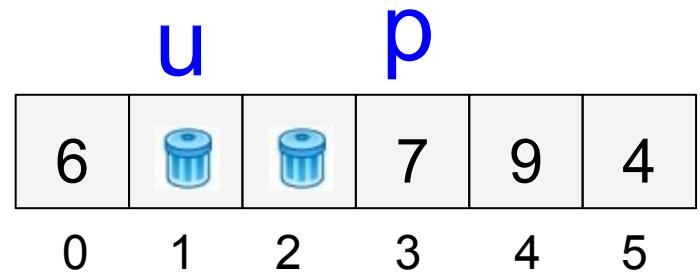
```
//Inserir(8)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



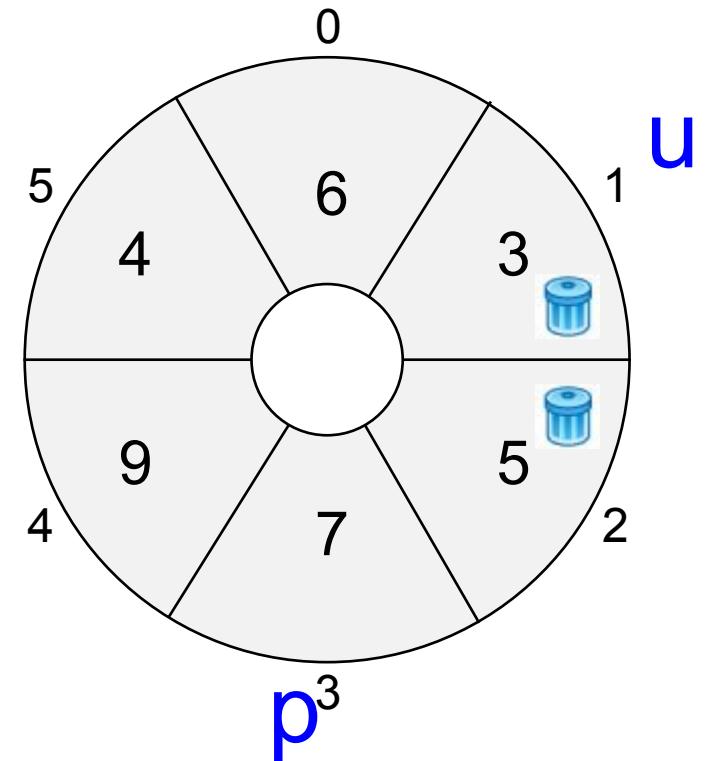
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



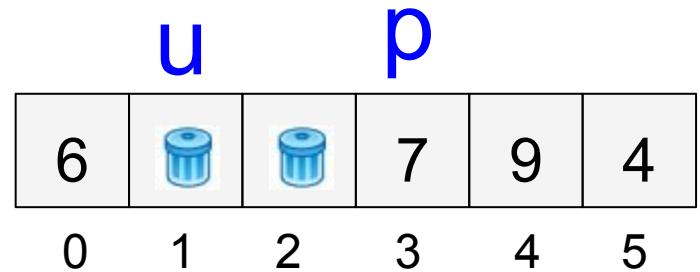
Algoritmo em Java

```
//Inserir(8)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");
    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
false: 1 + 1 % 6 == 3
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

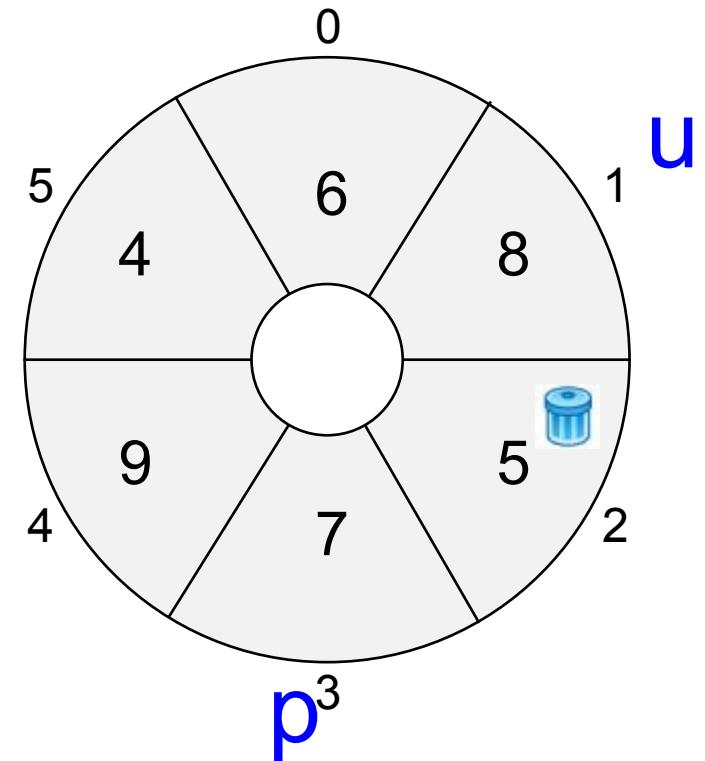


Algoritmo em Java

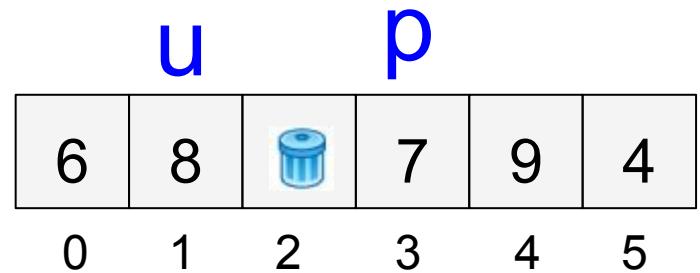
```
//Inserir(8)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

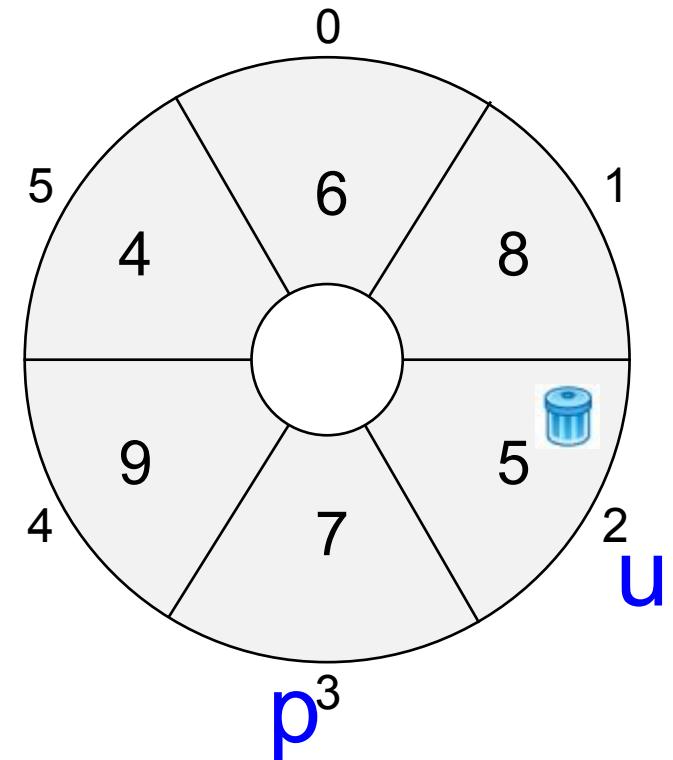


Algoritmo em Java

```
//Inserir(8)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



u p

6	8		7	9	4
0	1	2	3	4	5

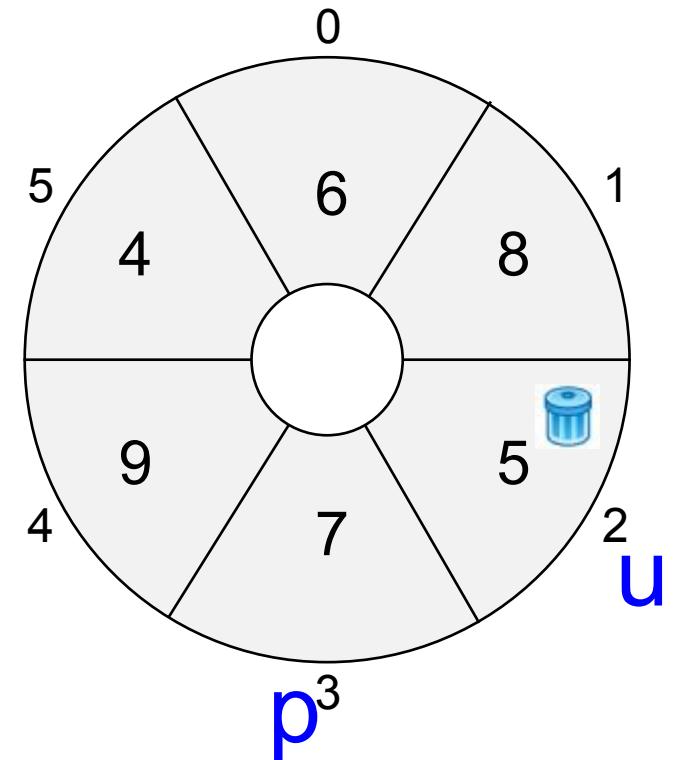
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), **I(8)**, M()

Algoritmo em Java

```
//Inserir(8)

void inserir(int x) throws Exception {
    if (((ultimo + 1) % array.length) == primeiro)
        throw new Exception("Erro!");

    array[ultimo] = x;
    ultimo = (ultimo + 1) % array.length;
}
```



u p

6	8		7	9	4
0	1	2	3	4	5

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), **I(8)**, M()

Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

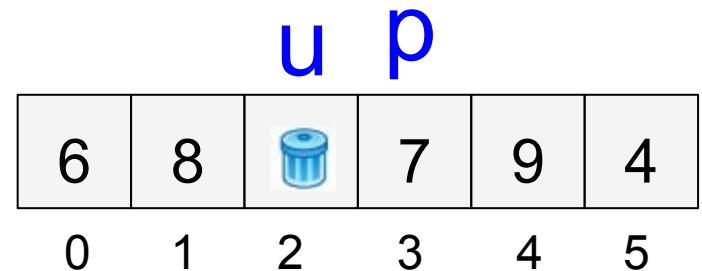
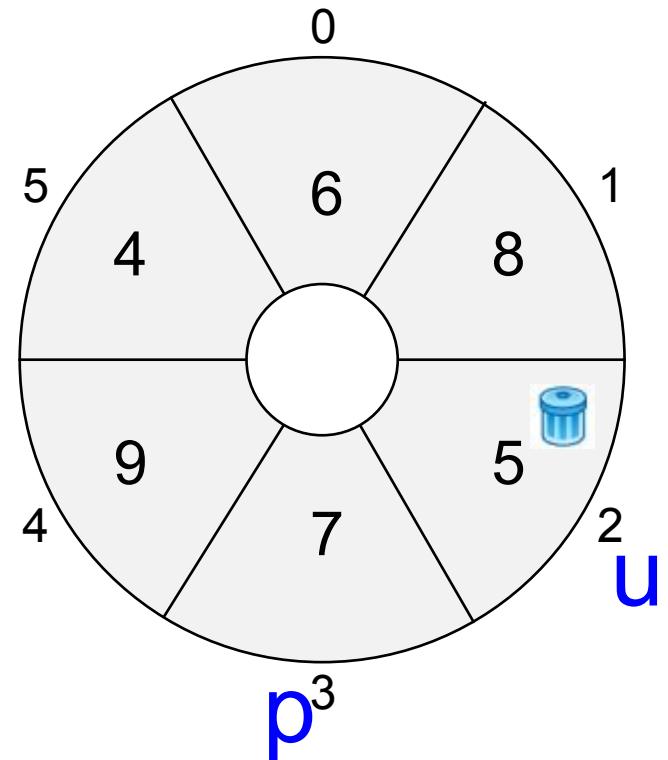
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

    System.out.println("]");
}

```

Tela:

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

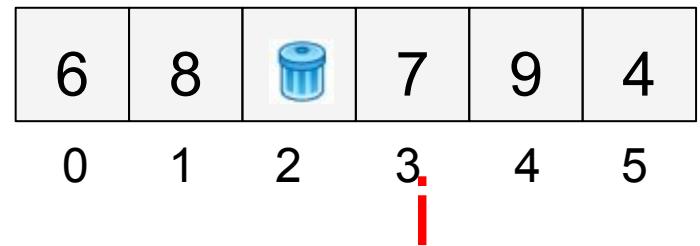
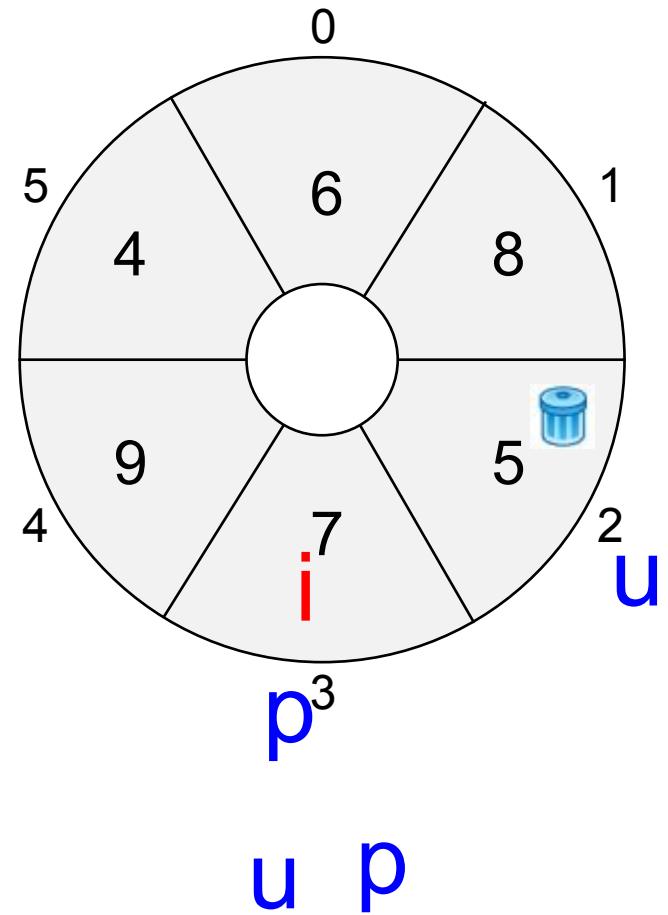
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

    System.out.println("]");
}

```

Tela:

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

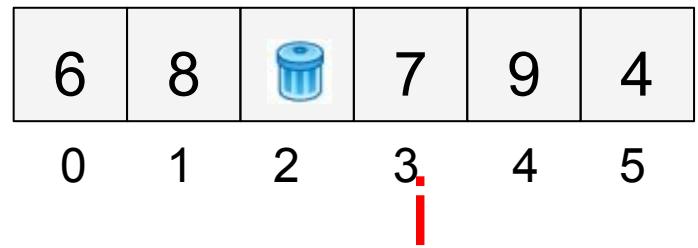
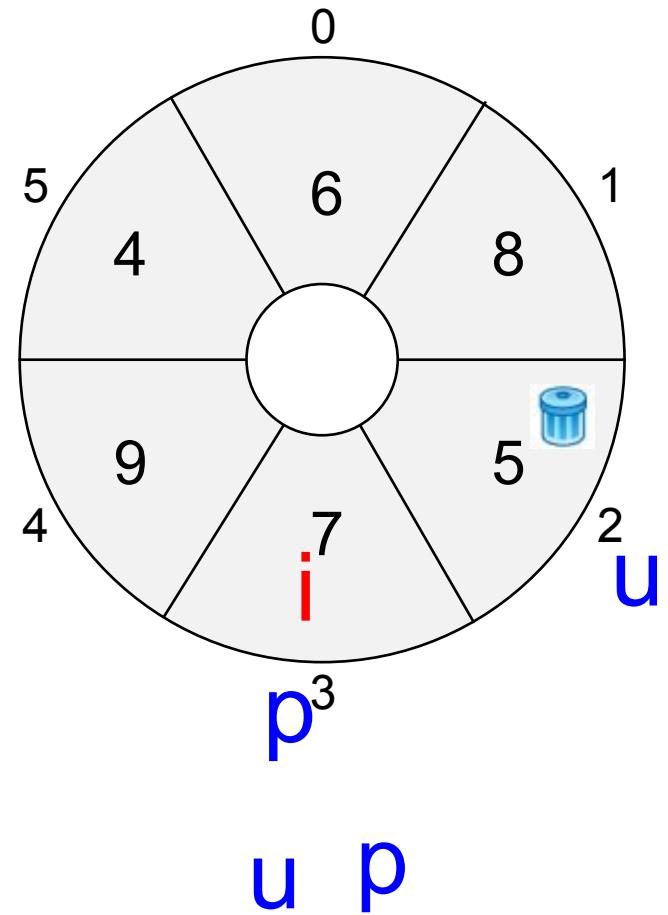
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

    System.out.println("]");
}

```

Tela: [

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

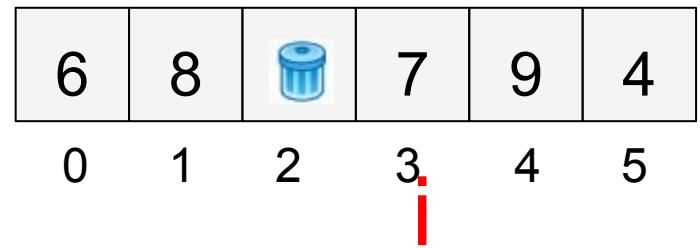
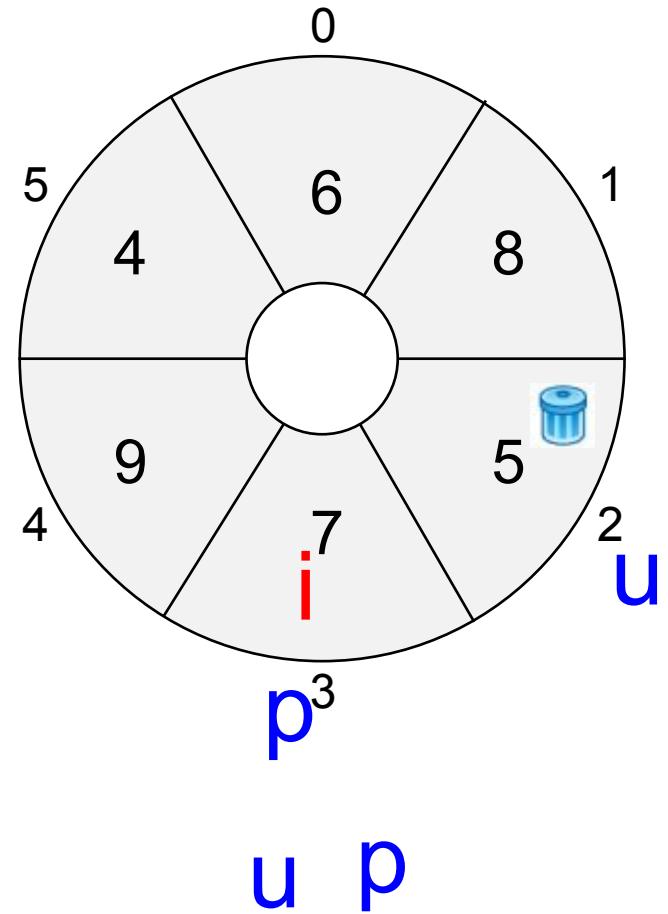
    System.out.println("]");
}

```

true: 3 != 2

Tela: [

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

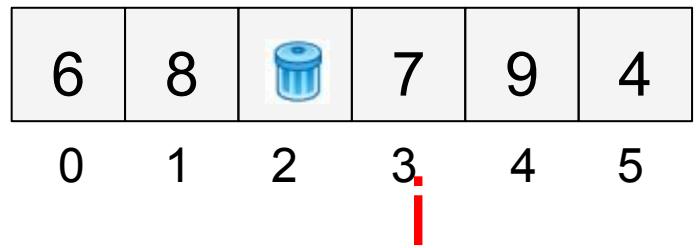
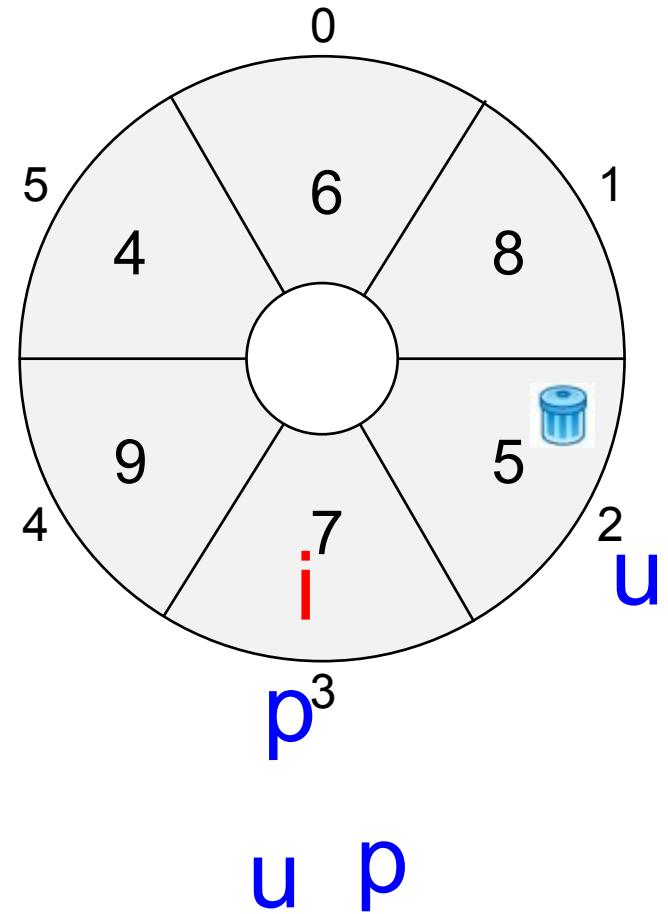
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

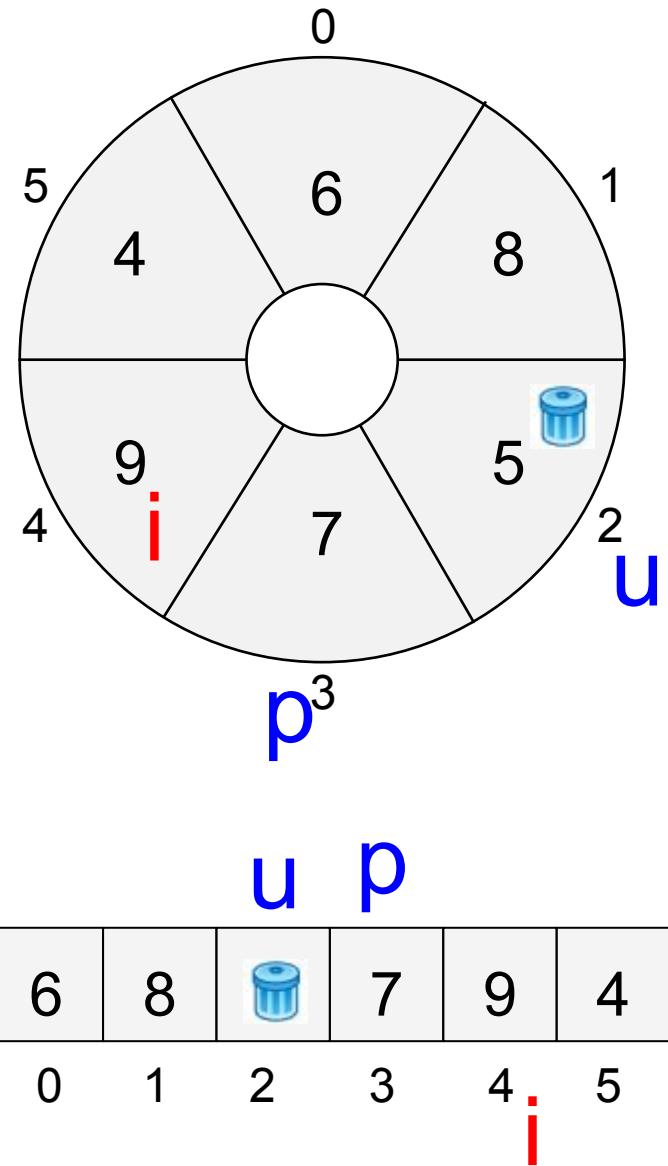
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

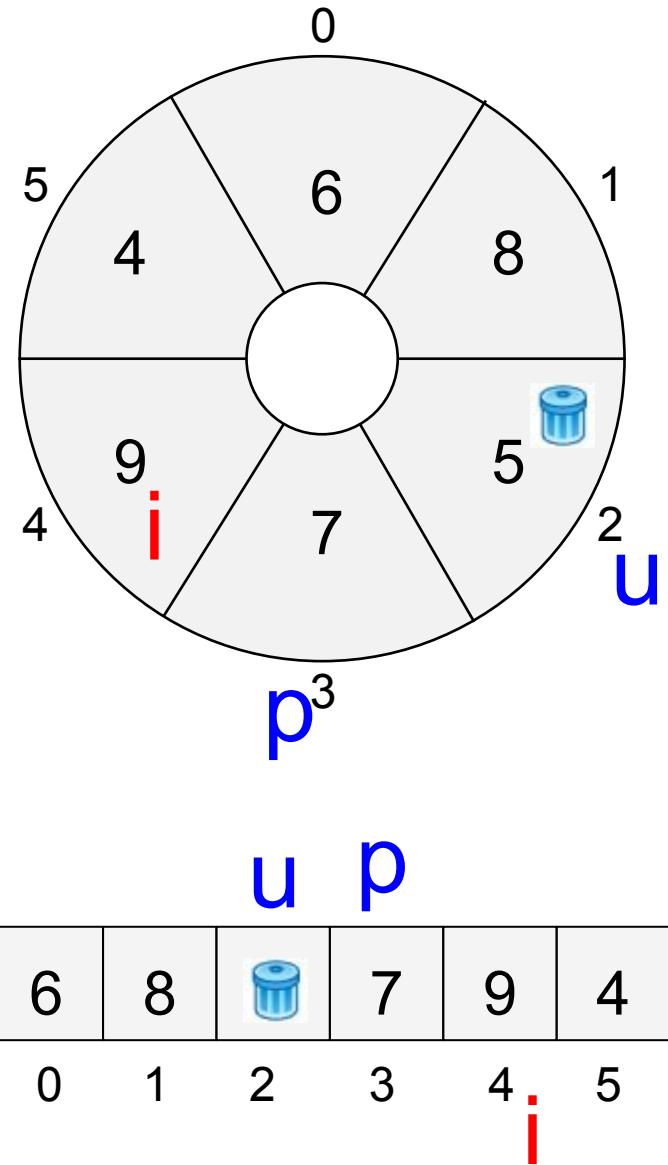
    System.out.println("]");
}

```

true: 4 != 2

Tela: [7

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

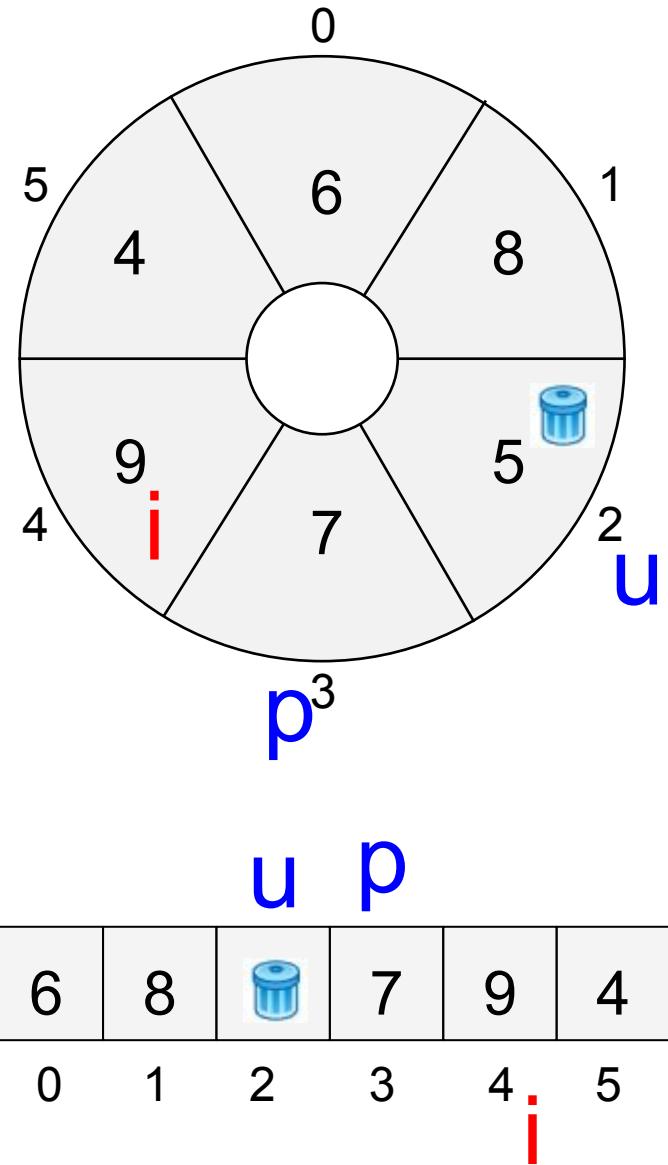
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

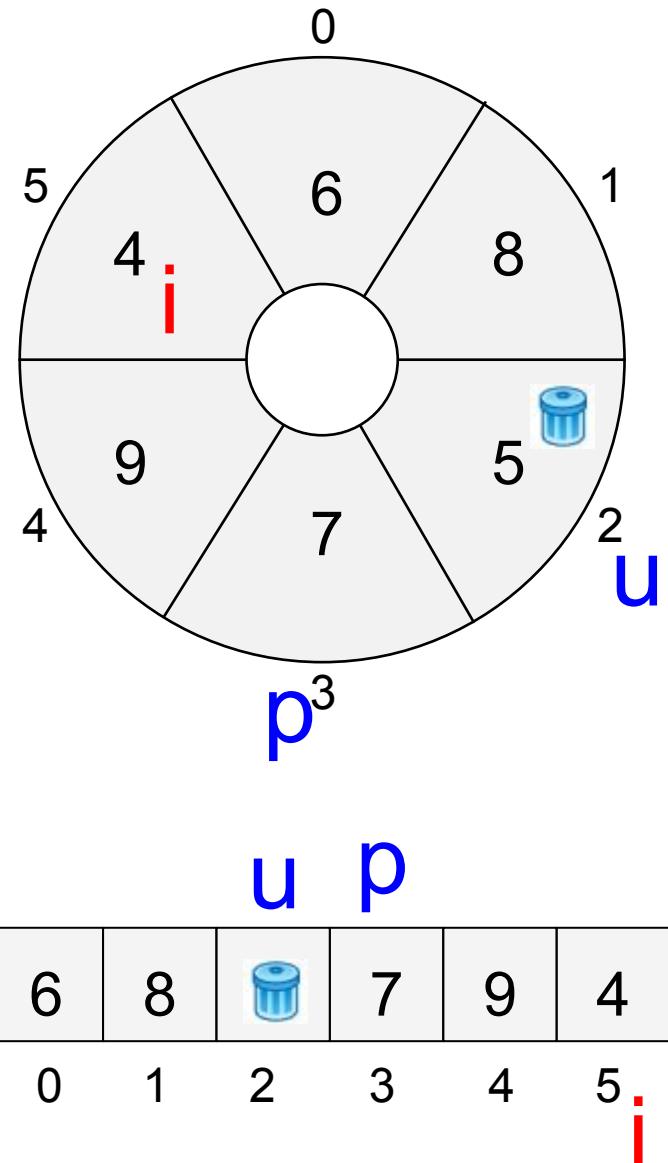
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

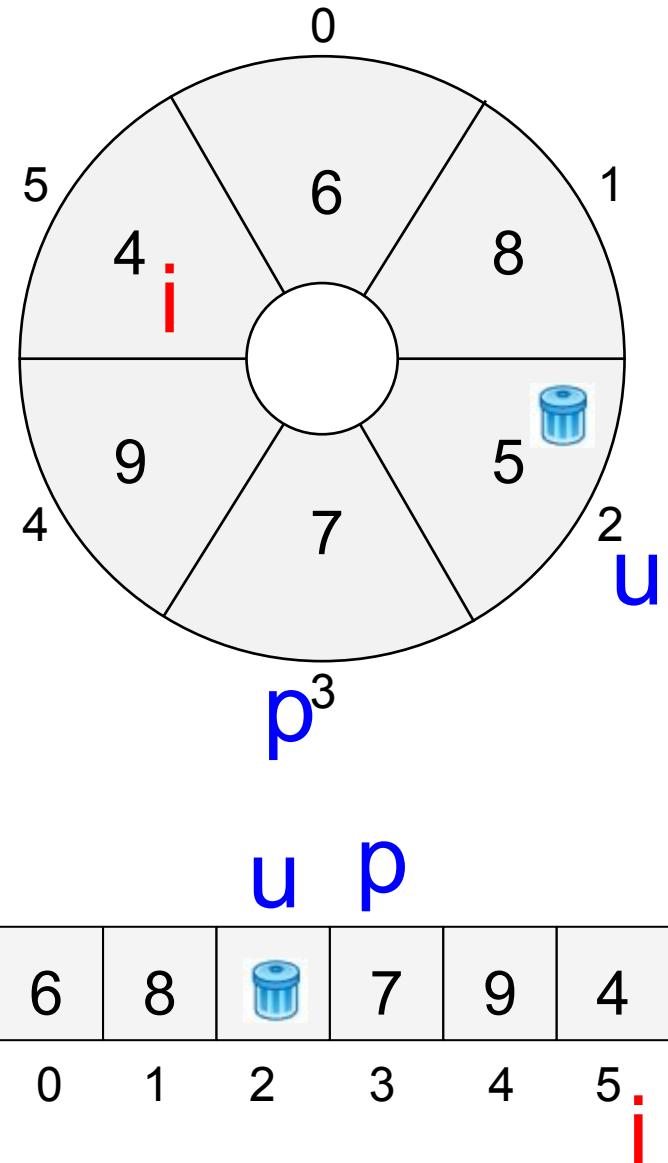
    System.out.println("]");
}

```

true: 5 != 2

Tela: [7 9]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

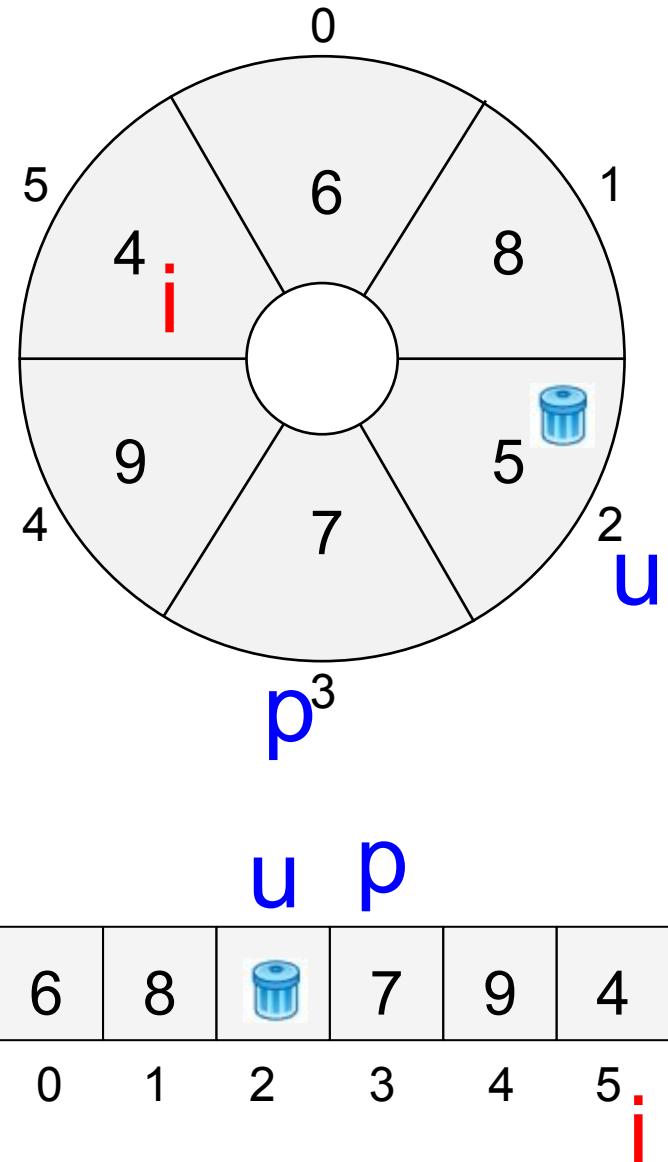
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9 4

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

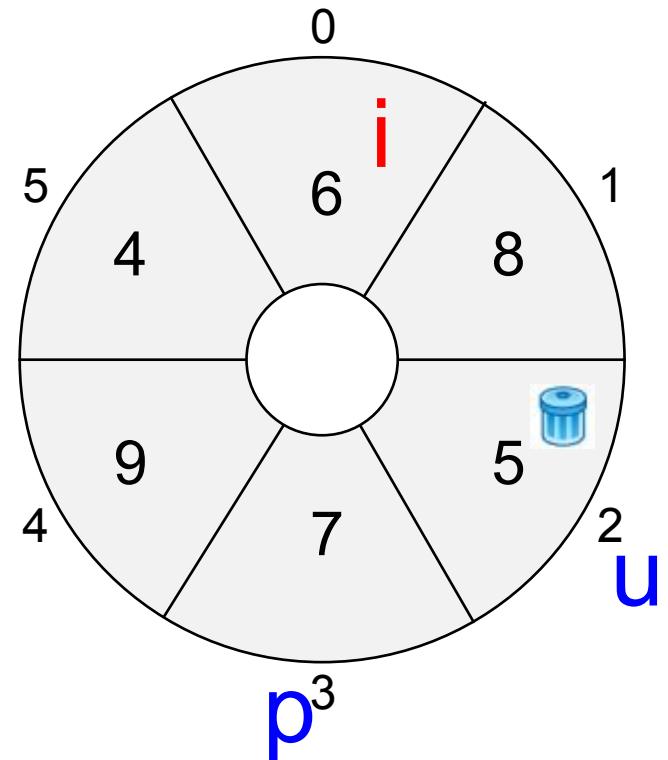
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9 4

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

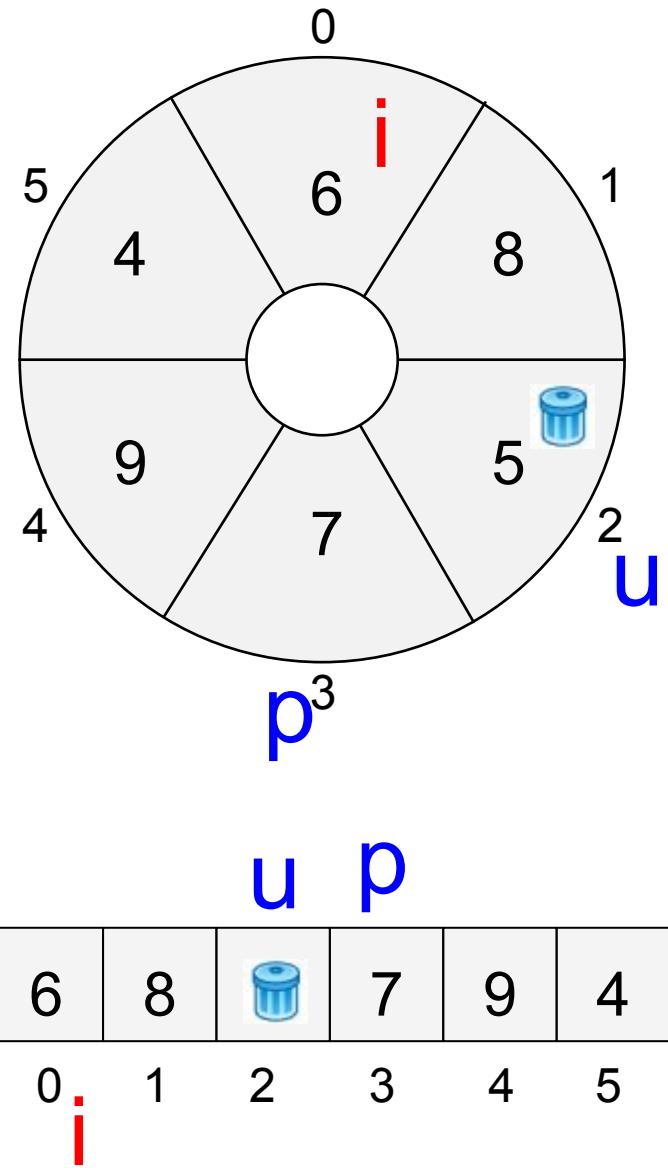
    System.out.println("]");
}

```

true: 0 != 2

Tela: [7 9 4

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

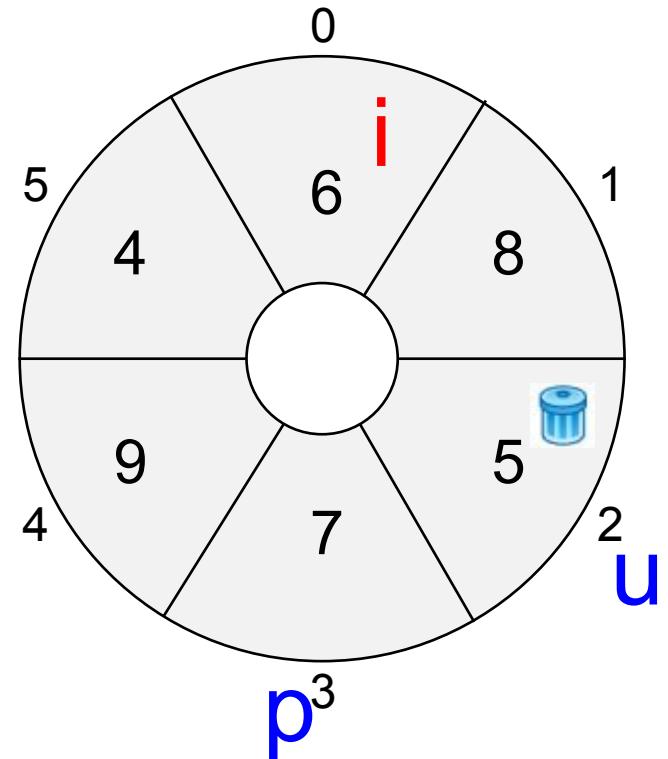
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9 4 6]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

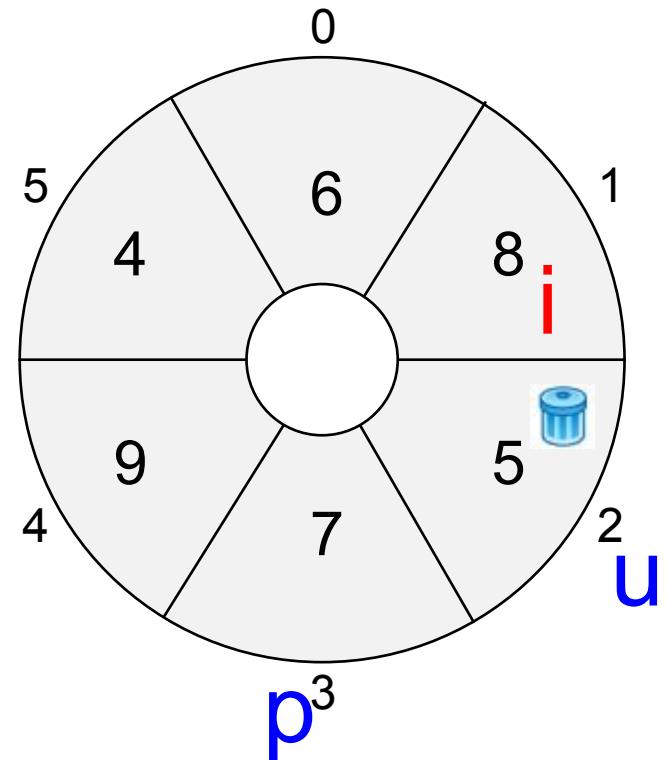
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9 4 6]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

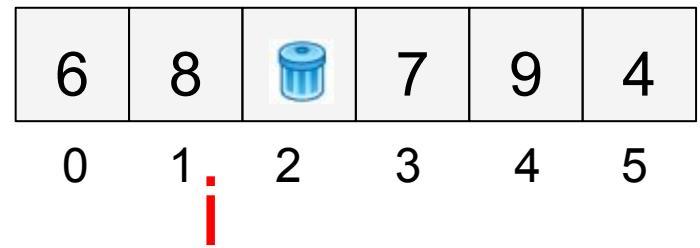
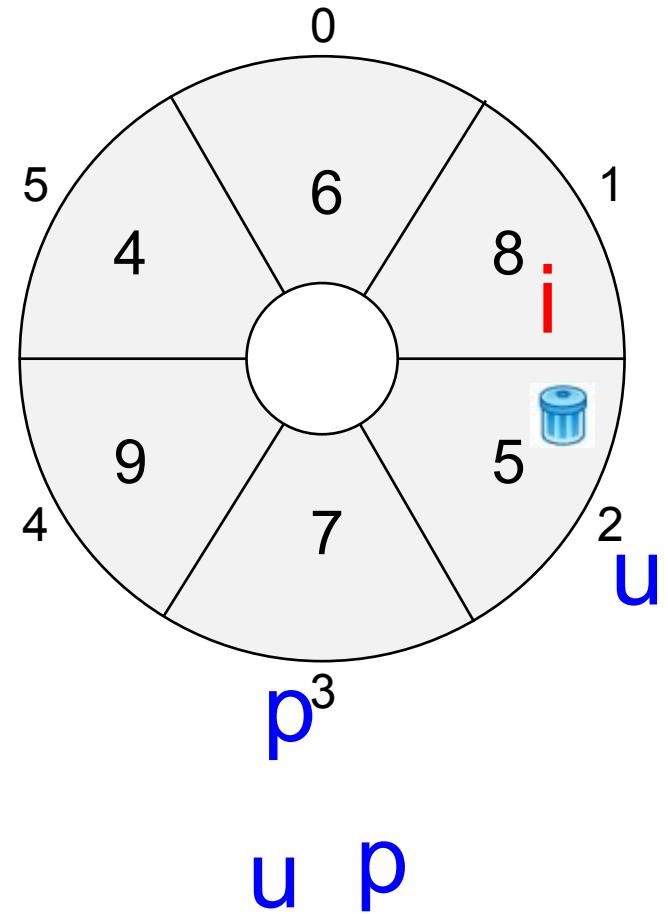
    System.out.println("]");
}

```

true: 1 != 2

Tela: [7 9 4 6]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

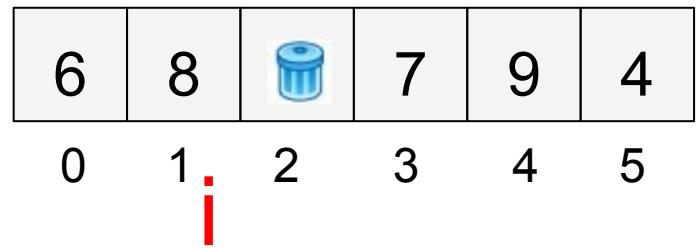
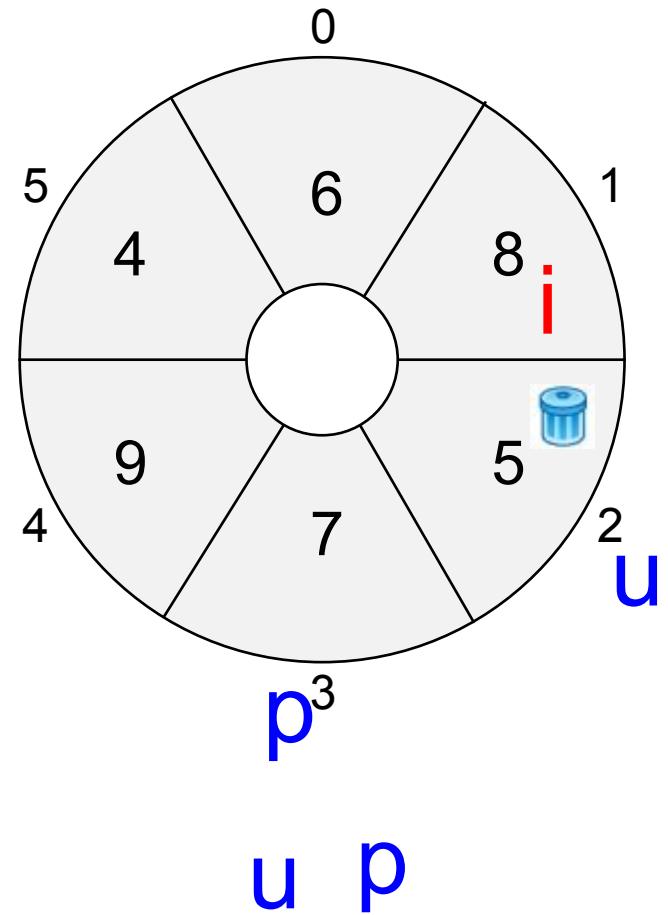
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

    System.out.println("]");
}

```

Tela: [7 9 4 6 8]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

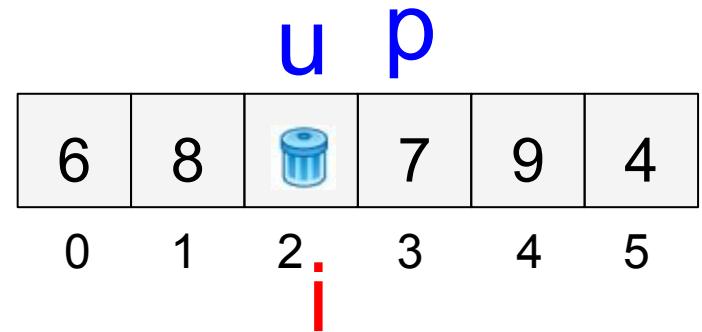
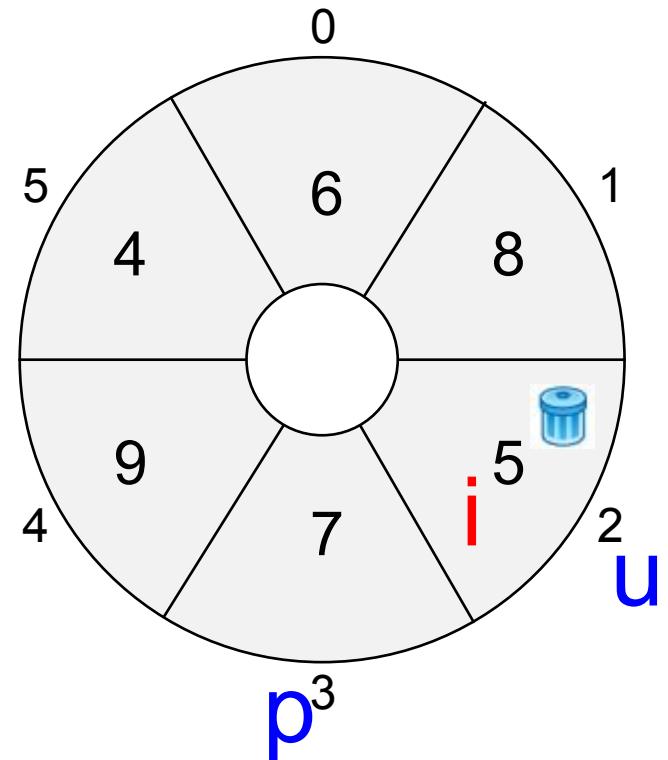
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9 4 6 8]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

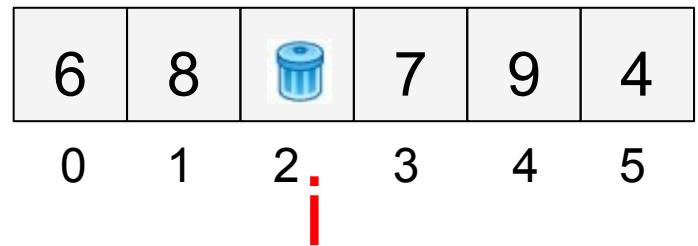
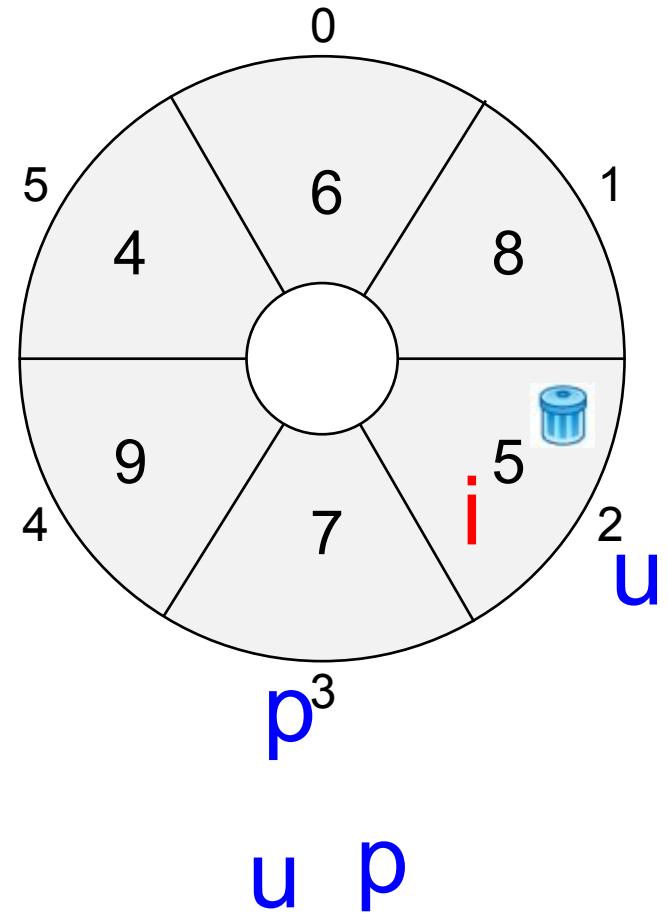
    System.out.println("]");
}

```

false: 2 != 2

Tela: [7 9 4 6 8

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

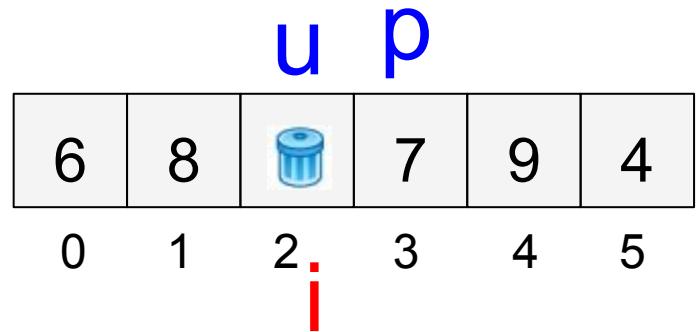
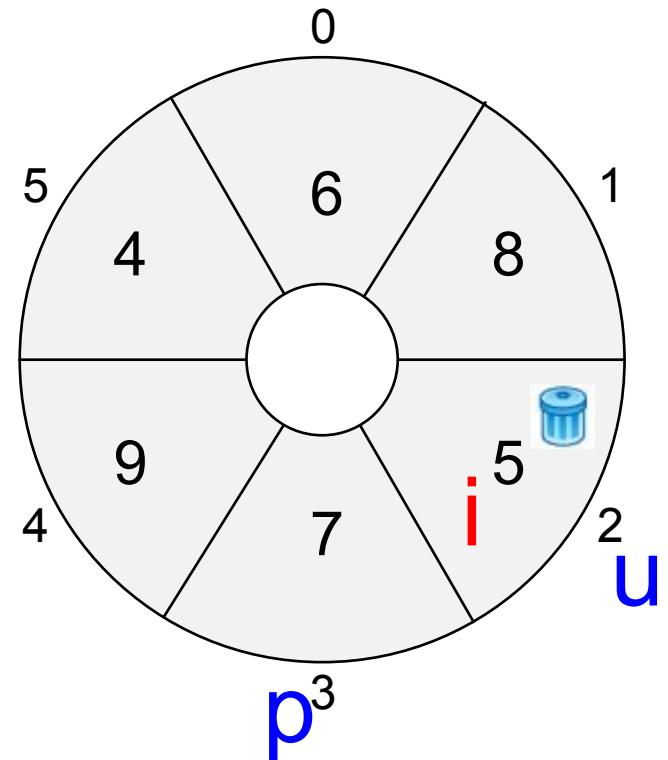
```

void mostrar() {
    int i = primeiro;
    System.out.print("[");
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }
    System.out.println("]");
}

```

Tela: [7 9 4 6 8]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em Java

```

void mostrar() {
    int i = primeiro;
    System.out.print("[");

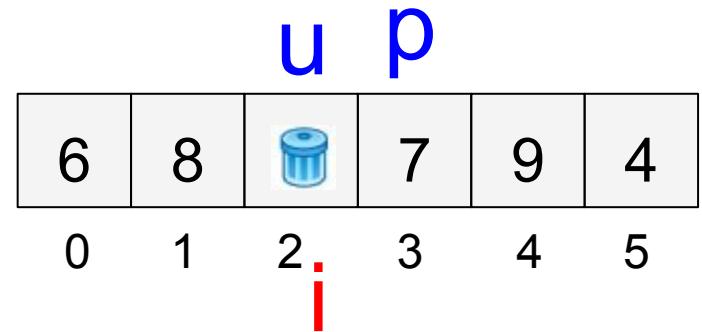
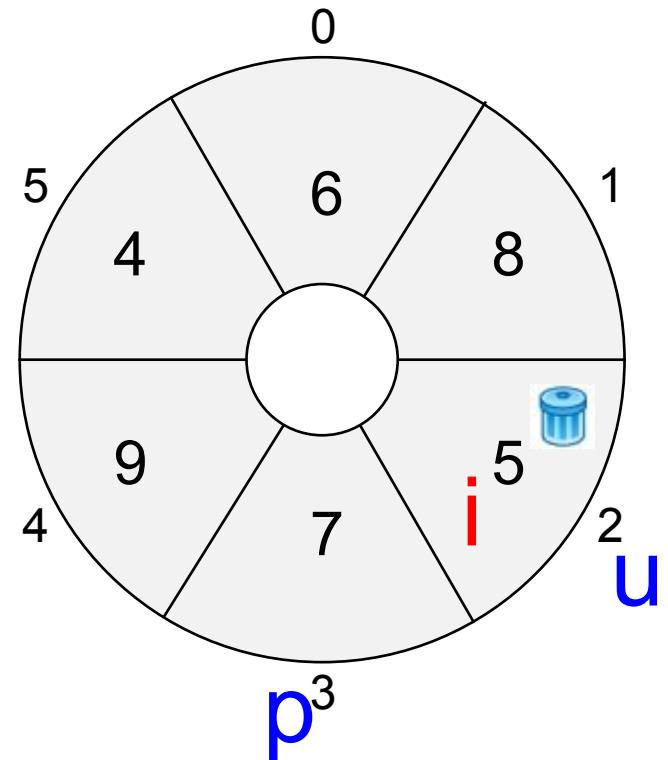
    while (i != ultimo) {
        System.out.print(array[i] + " ");
        i = (i + 1) % array.length;
    }

    System.out.println("]");
}

```

Tela: [7 9 4 6 8]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Exercício

- Implemente o método *boolean isVazio()*
- Implemente o método *void mostrarRec()* de forma recursiva
- Implemente o método *boolean pesquisar(int elemento)*
- Implemente o método *int retornaPos(int posicao)*
- Implemente a fila circular sem o “+1” do construtor e garantindo a quantidade de elementos solicitada

Agenda

- Conceitos Básicos
- Implementação Circular em Java
- **Implementação Circular em C** ←

Algoritmo em C

```
int array[MAXTAM+1];
```

```
int primeiro, ultimo;
```

```
void start() {  
    primeiro = ultimo = 0;  
}  
void inserir(int x) { ... }  
int remover() { ... }  
void mostrar() { ... }
```

Algoritmo em C

```
void inserir(int x) {
    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

```
int remover() {
    if (primeiro == ultimo)
        exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
}
```

```
void mostrar() {
    int i = primeiro;
    printf("[");
    while (i != ultimo) {
        printf(array[i] + " ");
        i = ((i + 1) % MAXTAM)
    }
    printf("]");
}
```